PAIRWISE ESTIMATING EQUATIONS FOR THE ANALYSIS OF LINKED DATA

by

Abel C. Dasylva

A dissertation submitted to the
Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

at

School of Mathematics and Statistics
Ottawa-Carleton Institute for Mathematics and Statistics
CARLETON UNIVERSITY

Ottawa, Ontario

August, 2018

©Copyright by Abel C. Dasylva, 2018

Abstract

In official statistics record linkage is an important activity, which consists in identifying records from the same individual in one or many files. It is used to combine data sources including administrative, survey or big data sources. In practice, record linkage is subject to linkage errors when it relies on quasi-identifiers, such as names and demographic variables, which are nonunique and recorded with errors. Accounting for these errors is an important but challenging problem. In this work, two methods are described for the primary analysis of such data, i.e. an analysis by someone with unfettered access to all the related micro-data and project information. Both solutions are estimating equation methods, which explicitly account for the uncertainty about the match status of record pairs and require the marginal distribution of a pair agreement vector. The first methodology is model-based and operates under the assumption of conditional independence between the pairs agreement vectors and the responses given the covariates. The second methodology uses a model-assisted estimating equation, which dispenses with the above assumption but requires reliable clerical-reviews.

Acknowledgments

Record linkage is a specialized activity that is little known in academia. I have become aware of its importance and related issues after joining Statistics Canada in the summer of 2010. My first project was the 2011 census overcoverage study, where probabilistic record-linkage was used extensively. It has been followed by many other projects and related activities that have allowed me to develop my expertise and identify relevant research problems. For this great opportunity, I am grateful to Karla Fox, Michel Hidirolou, Robert-Charles Titus, Christian Thibault, Dave Dolson, Claudia Sanmartin, Richard Trudeau, Abdelnasser Saidi, Yves Decady, and Scott McLeish. I am indebted to Prof. J.N.K. Rao for his interest in my work and his insights. I am also indebted to Statistics Canada Health Statistics Division for the data used in the empirical study. Finally, I would like to express my deep gratitude towards Prof. Sanjoy Sinha, for his guidance, patience and support all these years.

Dedication

I dedicate this thesis to my late mother Kalah Essih, to my father Joseph and to my loving wife and daughters, Diani, Yasminah and Madina.

Table of Contents

Abstract	ii
Acknowledgments	iii
Dedication	iv
Abbreviations	xv
1 Introduction	1
1.1 Nature of record linkage	1
1.2 Applications of record linkage	2
1.3 Linkage errors	3
1.4 Analytical challenge	4
1.5 Statement of the problem	5
1.6 Organization of the thesis	5
2 Background	7
2.1 Generalized linear models	7
2.1.1 Components	7
2.1.2 Some examples	8
2.1.3 Estimation and inference	9

2.2	Survival models	10
2.3	Probabilistic record linkage	11
2.3.1	Record linkage as hypothesis testing	11
2.3.2	Mixture models	14
2.3.3	Estimation of errors	18
2.3.4	Blocking	20
2.4	Analyzing linked data	21
2.4.1	Maximum likelihood	22
2.4.2	Estimating equations (EE)	24
2.4.3	Bayesian solutions	27
3	Pairwise EEs when linking registers	29
3.1	Overview	29
3.2	Notation	29
3.3	Assumptions	31
3.4	Conditional response distribution	32
3.4.1	Information from a block	32
3.4.2	Information from a single pair	37
3.5	Estimation procedures	41
3.5.1	Weighted Least Squares	43
3.5.2	Maximum composite likelihood	50
3.6	Large sample theory	57
3.7	Simulation study	61
3.7.1	General setup	61
3.7.2	Linear model	63
3.7.3	Logistic model	66

3.7.4	Survival model	68
3.7.5	Conclusions	69
4	Pairwise EEs when linking samples	72
4.1	Overview	72
4.2	Notation	72
4.3	Assumptions	73
4.4	Conditional response distribution	74
4.4.1	Information from a block	75
4.4.2	Information from a single pair	85
4.5	Estimation procedures	90
4.5.1	Weighted Least Squares	90
4.5.2	Maximum composite likelihood	98
4.6	Large sample theory	105
4.7	Simulation study	105
4.7.1	General setup	106
4.7.2	Linear model	106
4.7.3	Logistic model	106
4.7.4	Survival model	108
4.7.5	Conclusions	111
5	Model-assisted EEs	113
5.1	Introduction	113
5.2	Notation	114
5.3	Model-assisted estimators	116
5.4	Sampling design	120
5.5	Simulations	122

5.5.1	Setup	122
5.5.2	Results	126
5.6	Conclusions	128
6	Application	135
6.1	Introduction	135
6.2	Data	136
6.2.1	Canadian Mortality Database	136
6.2.2	Canadian Community Health Survey	138
6.3	Probabilistic linkage	139
6.3.1	Variables	140
6.3.2	Blocking criteria	140
6.3.3	Comparison vector	141
6.3.4	Mixture model	141
6.4	Analysis	142
6.5	Results	143
7	Conclusions	146
	List of References	153
	Appendix A Mathematical background	158
A.1	Stochastic orders of magnitude	158
A.2	Matrix derivatives	159
	Appendix B Code	160
B.1	Chapter 3	160
B.1.1	Linear regression	160

B.1.2	Logistic regression	183
B.1.3	Survival model	203
B.2	Chapter 4	219
B.2.1	Linear regression	219
B.2.2	Logistic regression	245
B.2.3	Survival model	267
B.3	Chapter 5	284
B.4	Chapter 6	300

List of Tables

1.1	Confusion matrix including the True Positives (TPs), True Negatives (TNs), False Negatives (FNs) and False Positives (FPs).	4
3.1	Performance under a linear model using linked data from two registers with a block size of $N_h = 2$ and a CMP threshold of 0.9.	64
3.2	Performance under a linear model using linked data from two registers with a block size of $N_h = 8$ and a CMP threshold of 0.9.	65
3.3	Performance under a linear model using linked data from two registers with a block size of $N_h = 2$ and a CMP threshold of 0.0.	65
3.4	Performance under a logit model using linked data from two registers with a block size of $N_h = 2$ and a CMP threshold of 0.9.	67
3.5	Performance under a logit model using linked data from two registers with a block size of $N_h = 8$ and a CMP threshold of 0.9.	67
3.6	Performance under a logit model using linked data from two registers with a block size of $N_h = 2$ and a CMP threshold of 0.0.	68
3.7	Performance under an exponential PHM using linked data from two registers with a block size of with $N_h = 2$ and a CMP threshold of 0.9.	69
3.8	Performance under an exponential PHM using linked data from two registers with a block size of with $N_h = 8$ and a CMP threshold of 0.9.	70

3.9	Performance under an exponential PHM using linked data from two registers with a block size of with $N_h = 2$ and a CMP threshold of 0.0.	70
4.1	Linear model when linking two samples with $N_h = 2$ and a CMP threshold of 0.9.	107
4.2	Linear model when linking two samples with $N_h = 2$ and a CMP threshold of 0.0.	107
4.3	Linear model when linking two samples with $N_h = 8$ and a CMP threshold of 0.9.	108
4.4	Logistic model when linking two samples.	109
4.5	Logistic model when linking two samples with $N_h = 2$ and a CMP threshold of 0.2.	109
4.6	Logistic model when linking two samples with $N_h = 8$ and a CMP threshold of 0.9.	110
4.7	Survival model when linking two samples.	111
4.8	Survival model when linking two samples with $N_h = 2$ and a CMP threshold of 0.0.	111
4.9	survival model when linking two samples with $N_h = 8$ and a CMP threshold of 0.9.	112
5.1	Agreement frequencies for Scenario 6 based on [1, Table 5.1].	128
5.2	Relative bias and CV for cell (0,0) for scenarios 1 through 3.	129
5.3	Relative bias and CV for cell (0,0) for scenarios 4 through 6.	129
6.1	Age-adjusted hazard ratios for mortality associated with selected health behaviours, by sex, respondents aged 20 or older to 2003 and 2005 Canadian Community Health Surveys linked to Canadian Mortality Database [2].	137
6.2	Allocation for 2000/2001 CCHS sample.	139

6.3	Estimated mixture parameters.	144
6.4	Estimated regression coefficients.	145

List of Figures

5.1	Box plot of the relative bias for cell (0,0) in scenario 1. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.	130
5.2	Box plot of the relative bias for cell (0,0) in scenario 2. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.	130
5.3	Box plot of the relative bias for cell (0,0) in scenario 3. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.	131
5.4	Box plot of the relative bias for cell (0,0) in scenario 4. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.	131
5.5	Box plot of the relative bias for cell (0,0) in scenario 5. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.	132
5.6	Box plot of the relative bias for cell (0,0) in scenario 6. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.	132

Abbreviations

CCHS	Canadian Community Health Survey
CMDB	Canadian Mortality Database
CMP	Conditional Match Probability
CoD	Cause of Death
CV	Coefficient of Variation
DD	Day of birth
EE	Estimating Equation
EM	Expectation Maximization
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
G-LINK	Generalized Linkage
GLM	Generalized Linear Model
GREG	Generalized Regression Estimator
HR	Health Region
HT	Horwitz-Thompson
IAR	Incorrect at Random
IID	Independent and Identically Distributed
LL	Lahiri-Larsen
LLN	Law of Large Numbers
LSE	Least Squares Estimator

MAR	Missing at Random
MCLE	Maximum Composite Likelihood Estimator
MCMC	Monte Carlo Markov Chain
ML	Maximum Likelihood
MM	Month of birth
MSE	Mean Squared Error
PHM	Proportional Hazards Model
PPV	Positive Predicted Value
PW	Pairwise
QL	Quasi-Likelihood
SDLE	Social Data Linkage Environment
SRS	Simple Random Sample
TN	True Negative
TP	True Positive
WLS	Weighted Least Squares
WLSE	WLS Estimator
YY	Year of birth

Chapter 1

Introduction

1.1 Nature of record linkage

Record linkage consists in identifying records that come from the same entity, e.g. a person or a business, in one or many files [1, 3, 4]. It is a multidisciplinary subject that goes by different names in other fields. It is known as *entity resolution* in the computer science and information retrieval literature. It is called *deduplication* when the goal is finding duplicates within a file. Record linkage is often confused with *statistical matching*; a form of imputation that aims at finding records from *similar* but distinct entities in another file.

In record linkage, two records are called *matched* if they actually refer to the same entity. Otherwise, they are called *unmatched*. This information is the *match status* of the record pair, which is a latent variable because it is seldom directly observed. In practice, the records' attributes (e.g., names, the birthdate and the mailing address with person files) are compared to decide whether the records are *linked*, i.e. *deemed* to refer to the same entity.

Making accurate linkage decisions is challenging because of the need to balance false positive errors and false negative errors. A false positive error occurs when two

unmatched records are linked. A false negative error occurs when two matched records are not linked. In that regard, record linkage is akin to a hypothesis testing problem for which there are well known solutions, including the probabilistic method by Fellegi and Sunter [5].

Linked pairs are typically collected in a linked data set that is generally viewed as the main output of the linkage process. However, other important outputs include the comparison outcomes of all generated record pairs, including those that are not linked. In this thesis, record linkage is broadly defined as the production of comparison outcomes for each pair in a given set, even if no actual linkage decision is made for some (or all) of the pairs.

1.2 Applications of record linkage

Record linkage is an important tool in epidemiology and official statistics. It has been used for creating richer data sets based on administrative sources and their combination with other sources, including censuses or surveys. The resulting data sets, called *linked data sets*, contain more variables than the original data sets. In epidemiology, examples abound [6, 7]. Linked data are also widely used in official statistics [8]. At Statistics Canada, record linkage has been used to produce many analytical data sets. For example, Wilkins et al. [9] have linked the Canadian Community Health Survey (CCHS) and hospital records to study the association between smoking and hospitalization for acute care. Another example is the linkage between the Canadian mortality database and the CCHS [2]. However, record linkage has also been used at other statistical agencies such as the Australian Bureau of Statistics. A good example is the Australian Census Longitudinal Dataset (ACLD) that is based on the linkage between a 5% sample of the 2006 population census and the 2011 census [10].

1.3 Linkage errors

Linking two files is straightforward when each file contains an identifier, such as the Social Insurance Number (SIN), which uniquely identifies each individual. The problem is more challenging when relying on quasi-identifiers such as names, demographic variables and addresses, which are nonunique and recorded with errors. Thus linkage errors occur, which include false negatives and false positives. A *False Negative* (FN) occurs when two matched records are not linked. A *False Positive* (FP) occurs when two unmatched records are linked. The probabilistic method [5] is designed to explicitly control the rates of linkage errors while minimizing the number of record pairs that are resolved manually. However, accurately measuring these rates is an important issue.

When linking with quasi-identifiers, the fundamental problem is the uncertainty about the match status of the different pairs. There are two ways to deal with this issue. The first strategy is to make a linkage decision for each pair, to estimate the related error rates and to use the estimated rates in the analysis. The different error measures include the False Positive Rate (FPR), the False Negative Rate (FNR) and the Positive Predicted Value (PPV). The FPR is the conditional probability that a pair is linked given that it is unmatched. The FNR is the conditional probability that a pair is not linked given that it is matched. The PPV is the conditional probability that a pair is matched given that it is linked. The information about linkage errors is summarized in a *confusion matrix*, i.e. a 2×2 matrix given the frequencies of the pairs according to their match status (matched or unmatched in the rows) and their link status (linked or not linked in the columns). The rates of linkage error may be estimated with a statistical model [5], clerical-reviews or both. The second strategy is to model the uncertainty about a pair match status and to directly use this model

Table 1.1: Confusion matrix including the True Positives (TPs), True Negatives (TNs), False Negatives (FNs) and False Positives (FPs).

	Linked	Unlinked
Matched	TP	FN
Unmatched	FP	TN

in the analysis, without making any linkage decision. The goal is to account for each record that is potentially matched to a given record, and not just the one to which it happens to be linked to.

1.4 Analytical challenge

The analysis of linked data is complicated by the occurrence of linkage errors, which are a source of bias and additional variance. This issue has been discussed in many previous studies, starting with Neter et al. [11]. Accounting for linkage errors is more or less difficult depending on the available information. This information is most abundant for a *primary analyst*, e.g. an employee or deemed-employee of a statistical agency, who has access to all the linkage micro-data, including each pair linkage weight, when the probabilistic method is used. This information also includes potential pairs that are not linked and any available clerical-review sample, i.e. a sample of pairs that are each known to be matched or unmatched. Although such a sample may greatly facilitate the estimation procedure, procuring it is often difficult and expensive, in practice. A secondary analyst typically accesses the linked data at a research data center, where the information is quite limited and the analytical challenge greatest. For example, the released information may be limited to the linked pairs and overall rates of linkage errors.

1.5 Statement of the problem

In this thesis, we look at regression problems for a primary analyst, where the analytical data set is based on the linkage of two data sources. Our objective is to explicitly account for the uncertainty about the match status of record-pairs using the available information, including their estimated (model-based) conditional match probability¹ or their actual match status, when clerical-reviews are available. In this process, we can use marginal models or joint models. A marginal model gives the joint distribution of the match status and comparison vector for a single pair, as is the case in classical probabilistic linkage [5]. By contrast, a joint model gives the joint distribution of the match statuses and comparison vectors for many pairs [12–16]. In this thesis, marginal models are preferred to joint models because they involve fewer assumptions and are supported by existing packages, including G-LINK and the R Record Linkage package. Marginal models also require much less computations than joint models, especially with large files.

1.6 Organization of the thesis

The following chapters are organized as follows. Chapter 2 provides some background on generalized linear models (GLMs), probabilistic record linkage and previous solutions for the analysis of linked data. Chapter 3 describes the proposed model-based pairwise method for a data set that is based on the linkage of two registers for the same finite population. Chapter 4 extends the pairwise method to the linkage of two samples. Chapter 5 describes the model-assisted method when there are resources for reliable clerical-reviews. Chapter 6 describes an empirical study where the Canadian

¹The conditional match probability is a by-product of Expectation-Maximization (E-M) procedures that are implemented in packages such as G-LINK, RELAIS and the R record linkage package

Mortality Database (CMDB) is linked to the 2000/2001 Canadian Community Health Survey (CCHS). Chapter 7 gives the conclusions.

Chapter 2

Background

2.1 Generalized linear models

Generalized linear models (GLMs) are a well-known generalization of linear models, which have been widely used and discussed by many authors, including McCullagh and Nelder [17] and Agresti [18]; the latter in the case of categorical variables. We hereafter provide a brief overview of these models, due to their importance in previous work on the analysis of linked data.

2.1.1 Components

A classical linear model considers a sample of independent observations $y_1, \dots, y_n$, where each observation y_i has a normal distribution with mean $E[y_i] = \mathbf{x}_i^\top \boldsymbol{\beta}$, based on fixed covariate values $\mathbf{x}_1, \dots, \mathbf{x}_n$. The GLM theory extends this model by allowing data with a nonnormal distribution, and a more general relationship of the form $g(E[y_i]) = \mathbf{x}_i^\top \boldsymbol{\beta} = \eta_i$ between the mean response and the covariates, for some *link function* $g(\cdot)$. A GLM is characterized by its three components, including the parameter η_i called *systematic component*, its link function $g(\cdot)$ and its *random component* that is given by the actual distribution of the response.

2.1.2 Some examples

GLMs include many well-known models, starting with the linear model where the link function is the identity.

Binary responses: There is the well-known logistic model with the logit link $g(\pi) = \log(\pi/(1 - \pi))$. However, alternatives include the probit $g(\pi) = \Phi^{-1}(\pi)$, where $\Phi(\cdot)$ is the normal cdf, the complementary log-log $g(\pi) = -\log(-\log(1 - \pi))$ and the log-log $g(\pi) = -\log(-\log(\pi))$ link functions. The logit link is often preferred. In practice it is common to aggregate a binary response with other responses that share the same covariates. Such responses are said to form a covariates class [17, section 4.1.2., pp. 99]. The GLM is then expressed in terms of these aggregated responses that have binomial distributions, if the original responses are independent. In this case, the variance of the aggregated responses correspond to the nominal binomial variance. *Overdispersion* occurs if the response variance exceeds the binomial variance due to a correlation among the original responses.

Polytomous responses: A response is polytomous if there are three or more response categories. It is further classified as an ordinal, interval, or nominal response, depending on whether the response categories are ordered. This is the case for an ordinal or interval response. An interval response is an ordinal response where a numerical score is assigned to each category. A response is nominal if there is no ordering of the categories, whether explicit or implied. As for binary responses, the original responses are usually aggregated if they share the same covariates. The aggregated responses are represented by vectors that have multinomial distributions, if the original responses are independent. Otherwise, overdispersion occurs and becomes manifest when the aggregated responses have a variance-covariance matrix that exceeds¹ the expected multinomial variance-covariance matrix.

¹in the ordering the symmetric positive definite matrices

Log-linear models: Log-linear models are used when the responses are counts. In this case, the $\log(\cdot)$ link is used and the responses or aggregated responses may be assumed to have a Poisson distribution.

2.1.3 Estimation and inference

A large part of the GLM theory has been devoted to the estimation of the regression coefficients β , when the response distribution is from the *exponential dispersion family* [18, Section 4.4, pp. 133]. However, the quasi-likelihood method is used for more general situations.

Estimation by the maximum likelihood: Consider a response variable y_i that follows the exponential family distribution

$$f(y_i; \theta_i, \phi) = \exp \left(\frac{y_i \theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi) \right),$$

where $a(\cdot)$, $b(\cdot)$ and $c(\cdot)$ are known functions and ϕ is the *dispersion parameter*. The two model parameters θ_i and ϕ are related to the mean response μ_i and the variance $\text{var}(y_i)$ through the following equations (see [17, Section 2.2, pp. 28-29] for a proof):

$$\begin{aligned} \mu_i &= b'(\theta_i), \\ \text{var}(y_i) &= b''(\theta_i) a(\phi). \end{aligned}$$

The log-likelihood has the form

$$l(\beta; \mathbf{y}) = \sum_{i=1}^n \frac{y_i \theta_i - b(\theta_i)}{a(\phi)} + \sum_{i=1}^n c(y_i, \phi)$$

This likelihood is also viewed as a function of the means $\mu_1, \dots, \mu_n$, given how θ_i

relates to μ_i . Let $\boldsymbol{\beta} = [\beta_1 \dots \beta_p]^\top$. Then the likelihood equations have the form

$$\sum_{i=1}^N \frac{(y_i - \mu_i) x_{ij}}{\text{var}(y_i)} \frac{\partial \mu_i}{\partial \beta_j} = 0, \quad j = 1, \dots, p.$$

The goodness of fit is measured by the deviance that is computed as

$$D(\mathbf{y}; \hat{\boldsymbol{\mu}}) = -2(L(\hat{\boldsymbol{\mu}}) - L(\mathbf{y}; \mathbf{y})),$$

where $L(\hat{\boldsymbol{\mu}})$ is the maximum likelihood under the postulated model and $L(\mathbf{y}; \mathbf{y})$ is the likelihood of a *saturated model* where each mean μ_i is viewed as a free parameter when maximizing the likelihood.

Estimation by the quasi-likelihood: When the response distribution is not of the required parametric form, the above likelihood equations remain unbiased for $\boldsymbol{\beta}$ provided the specified mean-variance relationship remains valid. This relationship is characterized by the function $\nu(\cdot)$ such that $\text{var}(y_i) = \phi\nu(\mu_i)$.

2.2 Survival models

Survival models deal with survival times that are characterized by censoring. They relate the likelihood of survival to individual covariates, of which some may be time-dependent. In these models, the hazard function $h(\cdot)$ plays an important role because it measures the instantaneous risk of death at time t . This hazard function may be expressed as a function of the survival time distribution in the form

$$h(t) = \frac{f(t)}{1 - F(t)} = \frac{f(t)}{S(t)},$$

where $f(t)$ is the probability density of the survival time, while $F(t)$ and $S(t) = 1 - F(t)$ are the cumulative distribution function and survivor function respectively. Proportional hazards models (PHM) form an important class of models, where the hazards ratio remains constant over time, for any two individuals that have no time-dependent covariates. In such models, the hazard function has the following specific form.

$$h(t) = h_0(t) \exp(\mathbf{x}^\top \boldsymbol{\beta}),$$

where $h_0(\cdot)$ is the baseline hazard function. A PHM can use a parametric baseline distribution or estimate this distribution in a nonparametric manner. When specifying the baseline, common alternatives include the exponential distribution (where $h(t)$ is constant), the Weibull distribution (where $h(t) = \alpha t^{\alpha-1}$) and the extreme-value distribution (where $h(t) = e^{\alpha t}$).

2.3 Probabilistic record linkage

Record-linkage is essentially a hypothesis testing problem. However, there are also many practical issues.

2.3.1 Record linkage as hypothesis testing

The probabilistic method views the record-linkage problem as that of testing a *simple hypothesis*; a problem thoroughly studied and solved in hypothesis testing theory. The general problem may be described as follows. Consider a random observation $x \sim f(\cdot; \theta)$, where the parameter θ is unknown in some space Θ and either equals to θ_0 or θ_1 . The observation is to be assigned to one of the two candidate distributions, $f(\cdot; \theta_0)$ or $f(\cdot; \theta_1)$, while avoiding errors that are of two kinds. A type I error occurs if the observation comes from $f(\cdot; \theta_0)$ but is assigned to $f(\cdot; \theta_1)$. A type II error occurs

if the observation comes from $f(\cdot; \theta_1)$ but is assigned to $f(\cdot; \theta_0)$. Thus we need to test the null hypothesis that $\theta = \theta_0$, against the alternative hypothesis that $\theta = \theta_1$:

$$H_0 : \theta = \theta_0 \text{ vs. } H_1 : \theta = \theta_1.$$

A given test is characterized by its *rejection region* R , which is defined as the subset of the space $\mathcal{X}$, where the null hypothesis is rejected. The performance of the test is measured by its *power function* that is defined by $\beta(\theta) = P_\theta(x \in R)$, where $P_\theta(\cdot)$ denotes the probability with respect to the distribution $f(\cdot; \theta)$. For a simple hypothesis, the test is a *size α test* if $\beta(\theta_0) = \alpha$. It is a *level α test* if $\beta(\theta_0) \leq \alpha$. Any given test is a trade-off between the type I and the type II errors because it is impossible to make both errors arbitrarily small. One way to make this trade-off is to minimize the type II error subject to an upper-bound α on the type I error. The solution to this constrained optimization problem is given by the Neyman-Pearson lemma, which may be understood as follows. Suppose that there is a size- α test with a rejection region of the form $R = \{x \text{ s.t. } f(x; \theta_1) > k f(x; \theta_0)\}$ where k is nonnegative. Then the test is uniformly most powerful (UMP) among all α -level tests, i.e., the test is such that $\beta(\theta) \geq \beta'(\theta)$ for $\theta \neq \theta_0$ and any other α -level test with power function $\beta'(\cdot)$. In other words, the test rejects the null hypothesis with a greater probability than any other α -level test, when this hypothesis is false.

In record-linkage, the observation is the comparison vector $\gamma = [\gamma^{(1)} \dots \gamma^{(K)}]^\top$ of a record-pair, where $\gamma^{(k)}$ is usually a categorical variable that indicates the level of agreement between two linkage variables. The candidate distributions are the conditional distribution of γ given that the pair is matched, denoted by $P(\cdot | M)$, and the conditional distribution of γ given that the pair is unmatched, denoted by $P(\cdot | U)$, where $P(\cdot | M)$ and $P(\cdot | U)$ are multinomial distributions corresponding to

one trial and different probabilities for the possible comparison vectors. Thus we have the following hypothesis testing problem:

$$H_0 : \gamma \sim P(\cdot|M) \text{ vs. } H_1 : \gamma \sim P(\cdot|U).$$

In record-linkage, the type I error is called false negative rate (FNR), while the type II error is called false positive rate (FPR). The relative importance of a type of error depends on the intended use of the linked data. When record-linkage is used to build a sampling frame, it is more critical to minimize the false negatives to avoid contacting the same respondent twice for the same survey. For analytical studies, the emphasis is often placed on the false positives. In that latter situation, the goal is minimizing the FPR while maintaining the FNR below a target α , e.g., 5%. Then a UMP α -level test has a rejection region of the form $R = \{\gamma \text{ s.t. } P(\gamma|M) < \tau P(\gamma|U)\}$ for $\tau > 0$. Thus a pair is rejected if the ratio $P(\gamma|M)/P(\gamma|U)$ is below the threshold τ that depends on α .

Fellegi and Sunter [5] have considered a more general problem, where the goal is neither to minimize the FNR nor the FPR, but to minimize resources that are available to review pairs and determine their match status. However, the optimization is constrained by specified levels for the FNR and FPR. In this case, there are three possible decisions for each pair, including accepting the pair as matched, rejecting the pair and subjecting the pair to a review. The solution is a double test of hypothesis with three regions including an acceptance region $D = \{\gamma \text{ s.t. } P(\gamma|M)/P(\gamma|U) > \tau_2\}$, a rejection region $R = \{\gamma \text{ s.t. } P(\gamma|M)/P(\gamma|U) < \tau_1\}$ and a clerical-review region (also known as *grey zone*) $P = \{\gamma \text{ s.t. } \tau_1 \leq P(\gamma|M)/P(\gamma|U) \leq \tau_2\}$, where $\tau_1 < \tau_2$. The pairs in the acceptance region are called *definitive pairs*, those in the rejection region are called *rejected pairs* and the remaining pairs are called *possible pairs*. However,

the implementation of this decision rule raises many practical issues, starting with the estimation of the conditional probabilities $P(\gamma|M)$ and $P(\gamma|U)$.

2.3.2 Mixture models

The estimation of the conditional probabilities $P(\gamma|M)$ and $P(\gamma|U)$ is a difficult problem where the match status of record-pairs is not directly observed. Unfortunately this is often the case when record-linkage is based on quasi-identifiers such as names, birthdates and addresses. Even when clerical-review is feasible, the costs and reliability of clerical decisions are important issues. Consequently, the conditional probabilities are estimated via a mixture model of the following general form $P(\gamma) = P(M)P(\gamma|M; \psi_M) + P(U)P(\gamma|U; \psi_U)$, where $\psi = (P(M), \psi_M, \psi_U)$ is the vector of all the unknown parameters. The simplest such models assume the conditional independence of the components of the comparison vector γ ; a condition that is mathematically expressed as follows:

$$P(\gamma|M) = \prod_{k=1}^K P(\gamma^{(k)}|M),$$

$$P(\gamma|U) = \prod_{k=1}^K P(\gamma^{(k)}|U).$$

Under this assumption, the ratio of probabilities $P(\gamma|M)/P(\gamma|U)$ is conveniently expressed as a product of the ratios for each variable, i.e., $\prod_{k=1}^K (P(\gamma^{(k)}|M)/P(\gamma^{(k)}|U))$. This is why the Fellegi-Sunter decision rule is usually expressed in terms of a pair linkage weight that is $w = w_1 + \dots + w_K$, where $w_k = \log(P(\gamma^{(k)}|M)/P(\gamma^{(k)}|U))$ is the weight for variable k .

Under the conditional independence assumption, the vector ψ_M comprises of the marginal conditional probabilities of $\gamma^{(1)}, \dots, \gamma^{(K)}$ given that a pair is matched, while

ψ_M comprises of the marginal conditional probabilities of $\gamma^{(1)}, \dots, \gamma^{(K)}$ given that a pair is unmatched. The estimation of these parameters may be based on an iterative expectation-maximization (EM) procedure as follows. Suppose we have a sample with n independent and identically distributed (IID) record-pairs that have the comparison vectors $\gamma_1, \dots, \gamma_n$. For $k = 1, \dots, K$, let $\hat{P}(M)$, $\hat{P}(\gamma^{(k)} | M)$ and $\hat{P}(\gamma^{(k)} | U)$ denote the Maximum Likelihood Estimates (MLEs). They satisfy the following maximum likelihood (ML) equations for the observed data [19]:

$$\begin{aligned}\hat{P}(M) &= \frac{1}{n} \sum_{i=1}^n \hat{P}(M | \gamma_i), \\ \hat{P}(\gamma^{(k)} | M) &= \frac{\sum_{i=1}^n I(\gamma_i^{(k)} = \gamma^{(k)}) \hat{P}(M | \gamma_i)}{\sum_{i=1}^n \hat{P}(M | \gamma_i)}, \\ \hat{P}(\gamma^{(k)} | U) &= \frac{\sum_{i=1}^n I(\gamma_i^{(k)} = \gamma^{(k)}) \hat{P}(U | \gamma_i)}{\sum_{i=1}^n \hat{P}(U | \gamma_i)},\end{aligned}$$

where

$$\hat{P}(M | \gamma_i) = \left(1 + \left(\frac{1}{\hat{P}(M)} - 1 \right) \frac{\hat{P}(\gamma_i | U)}{\hat{P}(\gamma_i | M)} \right)^{-1}.$$

The MLEs are computed iteratively as follows. For $k = 1, \dots, K$, let $\hat{P}^{(t)}(M)$, $\hat{P}^{(t)}(\gamma^{(k)} | M)$ and $\hat{P}^{(t)}(\gamma^{(k)} | U)$ denote the parameter estimates in iteration t . The estimates for iteration $t+1$ are computed in two steps. First proceed with the E-step where the conditional match probabilities are computed as

$$\hat{P}^{(t)}(M | \gamma_i) = \left(1 + \left(\frac{1}{\hat{P}^{(t)}(M)} - 1 \right) \frac{\hat{P}^{(t)}(\gamma_i | U)}{\hat{P}^{(t)}(\gamma_i | M)} \right)^{-1} \quad (2.1)$$

Next proceed with the M-step, where the parameter estimates are updated as

$$\begin{aligned}
 \hat{P}^{(t+1)}(M) &= \frac{1}{n} \sum_{i=1}^n \hat{P}^{(t)}(M | \gamma_i), \\
 [10pt] \hat{P}^{(t+1)}(\gamma^{(k)} | M) &= \frac{\sum_{i=1}^n I(\gamma_i^{(k)} = \gamma^{(k)}) \hat{P}^{(t)}(M | \gamma_i)}{\sum_{i=1}^n \hat{P}^{(t)}(M | \gamma_i)}, \\
 \hat{P}^{(t+1)}(\gamma^{(k)} | U) &= \frac{\sum_{i=1}^n I(\gamma_i^{(k)} = \gamma^{(k)}) \hat{P}^{(t)}(U | \gamma_i)}{\sum_{i=1}^n \hat{P}^{(t)}(U | \gamma_i)}.
 \end{aligned} \tag{2.2}$$

In practice, the conditional independence assumption may not be satisfied. However, the Fellegi-Sunter decision rule is robust to departures from this assumption, so long as these departures do not change the ordering of the pairs by their linkage weight. The conditional independence assumption is more problematic when computing model-based estimates of linkage errors. This limitation has motivated the study of more general models where the conditional distributions $P(\gamma|M)$ and $P(\gamma|U)$ incorporate interactions. In these models, the conditional distributions have the following general form:

$$\begin{aligned}
 P(\gamma|M) &= \frac{\exp(\mathbf{x}_{\gamma|M}^\top \boldsymbol{\beta}_M)}{\sum_{\gamma' \in \Gamma} \exp(\mathbf{x}_{\gamma'|M}^\top \boldsymbol{\beta}_M)}, \\
 P(\gamma|U) &= \frac{\exp(\mathbf{x}_{\gamma|U}^\top \boldsymbol{\beta}_U)}{\sum_{\gamma' \in \Gamma} \exp(\mathbf{x}_{\gamma'|U}^\top \boldsymbol{\beta}_U)},
 \end{aligned}$$

where Γ is the set of all possible comparison vectors, $\mathbf{x}_{\gamma|M}$ and $\mathbf{x}_{\gamma|U}$ are two covariates vectors associated with γ , while $\boldsymbol{\beta}_M$ and $\boldsymbol{\beta}_U$ are two unknown parameters. The covariates vectors $\mathbf{x}_{\gamma|M}$ and $\mathbf{x}_{\gamma|U}$ depend on the selected interactions for each conditional distribution. Let n_γ denote the frequencies of γ in the sample of pairs, and let $n_{\gamma|M}$ and $n_{\gamma|U}$ denote the corresponding frequencies among the matched and unmatched

pairs respectively. Also let n_M denote the number of matched pairs in the sample. Then the complete data ML equations may be written in the form:

$$\begin{aligned}\widehat{P}(M) &= \frac{n_M}{n}, \\ \sum_{\gamma \in \Gamma} \mathbf{x}_{\gamma|M} \left(\frac{n_{\gamma|M}}{n} - P(\gamma|M; \widehat{\beta}_M) \right) &= \mathbf{0}, \\ \sum_{\gamma \in \Gamma} \mathbf{x}_{\gamma|U} \left(\frac{n_{\gamma|U}}{n} - P(\gamma|U; \widehat{\beta}_U) \right) &= \mathbf{0}.\end{aligned}$$

Note that the last two equations above are ML equations for general multinomial distributions [18]. The corresponding observed data ML equations are as follows.

$$\begin{aligned}\widehat{P}(M) - \frac{E[n_M | [n_{\gamma'}]_{\gamma' \in \Gamma}; \widehat{\psi}]}{n} &= 0, \\ \sum_{\gamma \in \Gamma} \mathbf{x}_{\gamma|M} \left(\frac{E[n_{\gamma|M} | [n_{\gamma'}]_{\gamma' \in \Gamma}; \widehat{\psi}]}{n} - P(\gamma|M; \widehat{\beta}_M) \right) &= \mathbf{0}, \\ \sum_{\gamma \in \Gamma} \mathbf{x}_{\gamma|U} \left(\frac{E[n_{\gamma|U} | [n_{\gamma'}]_{\gamma' \in \Gamma}; \widehat{\psi}]}{n} - P(\gamma|U; \widehat{\beta}_U) \right) &= \mathbf{0},\end{aligned}$$

where

$$\begin{aligned}
E \left[n_M \middle| [n_{\gamma'}]_{\gamma' \in \Gamma}; \hat{\boldsymbol{\psi}} \right] &= n \sum_{\gamma \in \Gamma} P \left(M \middle| \gamma; \hat{\boldsymbol{\psi}} \right), \\
E \left[n_{\gamma|M} \middle| [n_{\gamma'}]_{\gamma' \in \Gamma}; \hat{\boldsymbol{\psi}} \right] &= n_{\gamma} \frac{P \left(M \middle| \gamma; \hat{\boldsymbol{\psi}} \right)}{\sum_{\gamma' \in \Gamma} P \left(M \middle| \gamma'; \hat{\boldsymbol{\psi}} \right)}, \\
E \left[n_{\gamma|U} \middle| [n_{\gamma'}]_{\gamma' \in \Gamma}; \hat{\boldsymbol{\psi}} \right] &= n_{\gamma} \frac{P \left(U \middle| \gamma; \hat{\boldsymbol{\psi}} \right)}{\sum_{\gamma' \in \Gamma} P \left(U \middle| \gamma'; \hat{\boldsymbol{\psi}} \right)}, \\
P \left(M \middle| \gamma; \hat{\boldsymbol{\psi}} \right) &= \left(1 + \left(\frac{1}{\hat{P}(M)} - 1 \right) \frac{P \left(\gamma \middle| U; \hat{\beta}_U \right)}{P \left(\gamma \middle| M; \hat{\beta}_M \right)} \right)^{-1}.
\end{aligned}$$

As before, the MLE $\hat{\boldsymbol{\psi}}$ may be estimated iteratively using an EM procedure. Let $\hat{\boldsymbol{\psi}}^{(t)}$ denote the estimate in iteration t . Compute the next estimate as the solution of the following system of equations.

$$\begin{aligned}
\hat{P}^{(t+1)}(M) - \frac{E \left[n_M \middle| [n_{\gamma'}]_{\gamma' \in \Gamma}; \hat{\boldsymbol{\psi}}^{(t)} \right]}{n} &= 0 \\
\sum_{\gamma \in \Gamma} \mathbf{x}_{\gamma|M} \left(\frac{E \left[n_{\gamma|M} \middle| [n_{\gamma'}]_{\gamma' \in \Gamma}; \hat{\boldsymbol{\psi}}^{(t)} \right]}{n} - P \left(\gamma \middle| M; \hat{\beta}_M^{(t+1)} \right) \right) &= \mathbf{0} \\
\sum_{\gamma \in \Gamma} \mathbf{x}_{\gamma|U} \left(\frac{E \left[n_{\gamma|U} \middle| [n_{\gamma'}]_{\gamma' \in \Gamma}; \hat{\boldsymbol{\psi}}^{(t)} \right]}{n} - P \left(\gamma \middle| U; \hat{\beta}_U^{(t+1)} \right) \right) &= \mathbf{0}
\end{aligned}$$

2.3.3 Estimation of errors

The accurate estimation of linkage errors is an important problem in probabilistic linkage for at least two reasons, of which the most obvious is the need to determine the thresholds in the decision rule. However accurate estimates of errors are also required

for any subsequent analysis of the linked data. Linkage errors may be evaluated using a mixture model, clerical-reviews or both [20]. The most favourable situation occurs when the mixture model is correctly specified. In this case, consistent estimators of the different error rates may be computed from the sample, without clerical-reviews. For a probabilistic linkage where a single threshold τ is used, let ϵ_I and ϵ_{II} denote the corresponding FNR and FPR, respectively. Then the following estimators are consistent:

$$\begin{aligned}\hat{\epsilon}_I &= \frac{\sum_{i=1}^n I\left(\hat{P}(\gamma_i|M) / \hat{P}(\gamma_i|U) < \tau\right) \hat{P}(M|\gamma_i)}{\sum_{i=1}^n \hat{P}(M|\gamma_i)} \\ \hat{\epsilon}_{II} &= \frac{\sum_{i=1}^n I\left(\hat{P}(\gamma_i|M) / \hat{P}(\gamma_i|U) \geq \tau\right) \hat{P}(M|\gamma_i)}{\sum_{i=1}^n \hat{P}(U|\gamma_i)}\end{aligned}$$

where $\hat{P}(\gamma_i|M)$, $\hat{P}(\gamma_i|U)$ and $\hat{P}(M|\gamma_i)$ are consistent estimators that are based on the specified mixture model.

When the mixture-model is misspecified, the linkage errors may be measured with a probabilistic clerical-review sample that must be optimized to reduce costs. One must also account for potential clerical errors. Dasylyva et al. [21] have described solutions for these two issues. Regarding the costs, they have used the model information to optimize the sample design or the estimator that is then model-assisted. For clerical-errors, the solution is based on repeated clerical-reviews for each sampled pair. The rates of clerical-errors are estimated with a latent class model, where it is assumed that that for the same pair, different reviewers make clerical errors that are conditionally independent given the pair match status and other pair or reviewer covariates.

2.3.4 Blocking

Blocking is the application of simple criteria to screen out the overwhelming majority of unmatched pairs in the Cartesian product of two files. Blocking typically requires few computations per record-pair. In G-LINK, Statistics Canada probabilistic linkage solution, the remaining pairs are called *potential pairs*. Blocking is an essential step in the early stages of the linkage process, especially when dealing with large files. By its very nature, it generates false negatives. Thus ideal blocking criteria should screen out the largest number of unmatched pairs, while keeping most matched pairs. There are many different blocking methods. The simplest solution uses exact agreement on a blocking key that may be derived from the original linkage variables. In this case, a record-pair meets the criterion if the two records have the same key value. Such a criterion partitions each input file into nonoverlapping blocks as assumed by Chambers [22]. In practice, more sophisticated blocking strategies are used. For example, to reduce the impact on the FNR, blocking may be based on a union of simple criteria, where each criterion is defined by an exact agreement on a specific key. Such a blocking strategy is unlikely to partition the records into nonoverlapping subsets in any input file. Blocking has a major impact on the distribution of record pairs and should be accounted for when estimating the mixture parameters and when measuring the linkage errors. This is especially true for the unmatched pairs that may significantly depart from the conditional independence assumption due to blocking, see Thibaudeau [23]. Indeed, let $P(\gamma) = P(M)P(\gamma|M) + P(U)P(\gamma|U)$ denote the distribution of pairs in the Cartesian product of two files, and let B denote the satisfaction of the blocking criteria by a given pair. Then each potential pair² is distributed according to $P(\gamma|B) = P(M|B)P(\gamma|M \cap B) + P(U|B)P(\gamma|U \cap B)$. With ideal blocking criteria, we have $P(B|M) \approx 1$ which implies $P(\gamma|M \cap B) \approx P(\gamma|M)$.

²A pair that meets the blocking criteria

Thus ideal blocking criteria are expected to preserve the distribution of matched pairs, including any conditional independence property. The situation is much different for unmatched pairs, where $P(B|U) \ll 1$ by design. Any unmatched pair that satisfies the blocking criteria is likely to be atypical among the unmatched pairs, because the satisfaction of these criteria is a rare event among them. Indeed, such a pair is likely to have an exceptionally large number of agreements when compared to a typical unmatched pair in the Cartesian product. This means that the blocking criteria are expected to significantly modify the unmatched distribution, including the loss of any conditional independence in the original unmatched pairs.

The evaluation of false negatives due to blocking is an important open issue in record-linkage. Herzog et al. [3] have suggested a capture-recapture method based on independent blocking criteria. However designing independent blocking criteria is difficult because of the limited number of linkage variables, of which some are correlated. Another challenge is the uncertainty about the match status of the pairs that are selected by the different criteria.

2.4 Analyzing linked data

Solutions for the analysis of linked data have been described for linear regression [22, 24–26], logistic regression [22, 27, 28] and contingency tables [27], for survival data and outcomes study [29–32] and for capture-recapture and population size estimation problems [13, 33, 34]. These solutions involve different methods including the maximization of a likelihood [27, 31], estimating equations [22, 24–26, 35–40], and Bayesian solutions using multiple imputations [13, 41, 42].

2.4.1 Maximum likelihood

The maximum likelihood method has been used by Chipperfield et al. [27], Hof and Zwinderman [43] and Hof et al. [31]. Chipperfield et al. [27] consider the analysis of a logistic model or contingency table using the links from the linkage of two files, where the first file is a census, and the second file is a census or sample. They propose a methodology that includes two separate adjustments for the linkage errors, which include the false positives and two kinds of false negatives. A false negative of the first kind occurs when a record has a matching record in the other file, no link to this particular record but other outgoing links that are all false positives. A false negative of the second kind occurs when a record has a matching record in the other file but no outgoing link. The first adjustment is based on the maximization of a likelihood that is computed over the the set of links and accounts for the false positives and the false negatives of the first kind. This likelihood also uses the match status of each link that is included in a probability clerical review sample. The links are assumed to be IID and incorrect at random (IAR), i.e. a link is a false positive independently of the actual response given the covariates. In the simplest case where each file is a census and each record is linked, the methodology by Chambers et al. [27] operates as follows. Let $\mathbf{x}_i$ denote the covariates for record i in the first file and z_j denote the response for record j in the second file. Suppose that for $i = 1, \dots, n$, record i is linked to record j_i in the second file, such that we have n links $(1, j_1), \dots, (n, j_n)$. For each link, the complete data include the covariates $\mathbf{x}_i$, the observed response z_{j_i} , the actual response y_i and the match status m_{ij_i} . For each link, the observed data include the covariates $\mathbf{x}_i$ and the observed response z_{j_i} . For some links, the observed data further include the match status m_{ij_i} and the actual response y_i . The match status m_{ij_i} is observed only if the link is included in the clerical-review sample. As for the actual response y_i , it is observed only when the link is in the clerical sample and a true

positive. The relevant parameters are estimated by maximizing the likelihood for the observed data. For each link where the observed data and complete data differ, this likelihood is computed as the conditional expectation of the complete data likelihood given the observed data. An iterative EM procedure is used, where a key input is the estimated conditional probability that a link is matched given the covariates $\mathbf{x}_i$ and the observed response z_{ji} , i.e. $P(m_{ij_i} | \mathbf{x}_i, z_{ji})$. This critical parameter is estimated separately using the clerical-review sample. The second adjustment accounts for the second kind of false negatives by reweighting the links.

Hof and Zwinderman [43] consider the estimation of a GLM with a linked data set that is based on a probabilistic record linkage. They propose a method for the joint estimation of the linkage parameters and regression parameters, which does not require a linkage decision for each pair but instead uses all the pairs that satisfy the blocking criteria. However, two critical assumptions are made. The first assumption is the conditional independence of the analytical variables and the comparison vector³ given the match status, in each record-pair. The second assumption is that each file is comprised of records from distinct independent individuals, such that no two records are from the same individual. Consider the record pair (i, j) , where record i comes from the file with the covariates and record j is from the file with the responses. Let $\mathbf{x}_i$, z_j , γ_{ij} denote the corresponding covariates, observed response, and comparison vector, respectively. Let $f_{y|x}(\cdot; \boldsymbol{\beta})$ denote the conditional distribution of the response given the covariates, $f_x(\cdot)$ denote the marginal distribution of the covariates, and $\boldsymbol{\psi}$ the vector of parameters for the linkage model, which further assumes the conditional independence of the linkage variables. The likelihood of a record pair is the following

³The vector that results from the comparison of the linkage variables

mixture:

$$\begin{aligned} f(z_j, \mathbf{x}_i, \gamma_{ij}) &= P(M)P(\gamma_{ij} | M; \boldsymbol{\psi}) f_{y|x}(z_j | \mathbf{x}_i; \boldsymbol{\beta}) f_x(\mathbf{x}_i) + \\ &P(U)P(\gamma_{ij} | U; \boldsymbol{\psi}) f_y(z_j) f_x(\mathbf{x}_i), \end{aligned}$$

where M is the event that the pair is matched, U is the event that it is unmatched, and $f_y(\cdot)$ is the unconditional response distribution. The parameters $\boldsymbol{\beta}$ and $\boldsymbol{\psi}$ are estimated with an expectation maximization procedure that maximizes the total log-likelihood over all the pairs, i.e. $\log L = \sum_{i,j} \log f(z_j, \mathbf{x}_i, \gamma_{ij})$, as if the record-pairs were independent. For computational efficiency, the total is actually limited to the record-pairs that satisfy the blocking criteria. The solution has been applied for a linear regression and a logistic regression on pregnancy data. In subsequent work [31], Hof et al. extend this methodology to parametric proportional hazards models (PHMs).

2.4.2 Estimating equations (EE)

There are essentially two families of EE-based solutions. The first family starts with the work by Scheuren and Winkler [24, 25] and continues with the study by Lahiri and Larsen [26]. The second family originates with the study by Chambers [22].

First family of EEs: In a first paper, Scheuren and Winkler [24] consider the problem of linear regression with pairs that come from a probabilistic linkage including some that are not linked. They propose a bias-correction method that estimates the bias of the naive least-squares estimator, by exploiting the information from the linkage model. However, the resulting (bias-corrected) estimator has some residual bias. In a second related paper, Scheuren and Winkler [24] propose a robust version of their

estimator to deal with outliers. Lahiri and Larsen [26] also address the problem of linear regression and propose an improved unbiased estimator. The solution applies when the data is based on the linkage of two registers that are free of duplicates and such that each record has a matching record in the other file. In order to better describe this solution, consider two registers for a population of N individuals. In the first file, let $\mathbf{X} = [\mathbf{x}_1^\top \dots \mathbf{x}_N^\top]$ denote the matrix of fixed covariates, where $\mathbf{x}_i$ denotes the covariates in record i . Although it is not observed directly, let $\mathbf{y} = [y_1 \dots y_N]^\top$ denote the vector of all the actual responses, where y_i is the response for record i from the covariates file. In the second file, let $\mathbf{z} = [z_1 \dots z_N]^\top$ denote the vector of all the responses, where z_j is the observed response in record j . Let m_{ij} denote the match status of the pair (i, j) , i.e. the indicator variable equal to 1 if the pair is matched and equal to 0 otherwise. Also let $\mathbf{M} = [m_{ij}]_{1 \leq i, j \leq N}$ denote the match matrix that is a permutation. Assuming the independence of $\mathbf{y}$ and $\mathbf{M}$, Lahiri and Larsen [26] note the identity

$$E[\mathbf{z}] = E[\mathbf{M}^\top] \mathbf{X} \boldsymbol{\beta}$$

and propose the corresponding least-squares estimator (LSE) that is unbiased.

$$\hat{\boldsymbol{\beta}} = (\mathbf{W}^\top \mathbf{W})^{-1} \mathbf{W}^\top \mathbf{z},$$

where $\mathbf{W} = E[\mathbf{M}^\top] \mathbf{X}$ is a key parameter that is based on the expected match matrix. Lahiri and Larsen [26] outline how this matrix may be estimated from the record-linkage mixture model. They also propose a bootstrap method for computing the variances, while accounting for the additional variance that comes from the estimation of the linkage parameters. Lahiri and Law [39] have extended this solution to GLMs under the same assumption, i.e. the independence of $\mathbf{y}$ and $\mathbf{M}$, which has the

following key implication:

$$E[\mathbf{z}] = E[\mathbf{M}^\top] \boldsymbol{\mu}, \quad (2.3)$$

where $\boldsymbol{\mu} = E[\mathbf{y}]$. Hof and Zwiderman [44] describe other extensions for logistic regression, and covariates that are distributed over two or more files. They also propose weighted least-squares estimators (WLSEs) for linear regression and logistic regression models.

Second family of EEs: The second EE family originates with Chambers [22]. In his study, Chambers still considers that the analytical file is produced by linking two registers. However, the setup differs from that considered by Lahiri and Larsen [26] in many respects, including the regression problem that is more general and includes GLMs as a special case. Another important difference is that the registers are actually linked, in the sense that a linkage decision is made for each record-pair, such that any record is linked to exactly one record in the other register. In order to elaborate on this solution, let l_{ij} denote the linkage decision for the pair (i, j) , i.e. the indicator variable set to 1 if the pair is linked, and let $\mathbf{L} = [l_{ij}]_{1 \leq i, j \leq N}$ denote the link matrix, which is also a permutation matrix. Also define the linkage error matrix $\widetilde{\mathbf{M}} = \mathbf{M}\mathbf{L}^\top$ and the transformed observed responses $\widetilde{\mathbf{z}} = \mathbf{L}\mathbf{z}$. When the actual responses $\mathbf{y}$ and the linkage error matrix $\widetilde{\mathbf{M}}$ are independent⁴, i.e. the linkage is IAR, Chambers notes that

$$E[\widetilde{\mathbf{z}}] = E[\widetilde{\mathbf{M}}^\top] \boldsymbol{\mu},$$

where $\boldsymbol{\mu} = E[\mathbf{y}]$. As a consequence, he proposes an estimation procedure based on

⁴The covariates are still considered fixed.

the estimating equation

$$\mathbf{A}(\mathbf{X}) \left(\tilde{\mathbf{z}} - E \left[\widetilde{\mathbf{M}}^\top \middle| \mathbf{X} \right] \boldsymbol{\mu} \right) = 0,$$

where $\mathbf{A}$ is a suitable multiplier matrix. For linear regression, the choice $\mathbf{A} = \mathbf{X}^\top$ produces the LSE

$$\hat{\boldsymbol{\beta}} = \left(\widetilde{\mathbf{W}}^\top \widetilde{\mathbf{W}} \right)^{-1} \widetilde{\mathbf{W}}^\top \mathbf{z},$$

where $\widetilde{\mathbf{W}} = E \left[\widetilde{\mathbf{M}}^\top \right] \mathbf{X}$. Note that the estimator by Lahiri and Larsen is the special case of Chambers' solution, when the link matrix is the identity. The expected linkage error matrix $E \left[\widetilde{\mathbf{M}} \right]$ is a key parameter that is difficult to estimate without clerical reviews. Chambers also proposes WLSE including the best linear unbiased estimator (BLUE) and the empirical BLUE (EBLUE). This solution has been extended in many directions including

- the linkage of a sample to a register [36],
- finite population inference [35], and
- the probabilistic linkage of three files, where a first file contains the responses, while the remaining files contain different subsets of covariates [37].

2.4.3 Bayesian solutions

Larsen [41] considers the problem of linear regression with linked data, when the goal is producing point and variance estimates, while accounting for all sources of variability, including the pairs match statuses and the estimated linkage parameters. A Bayesian multiple imputation solution is described where several sets of linkage parameters and links are drawn from the posterior distribution. For each set, the

regression coefficients are estimated and the results from the different sets are combined to produce overall point and variance estimates. However, the solution does not account for blocking or linkage constraints⁵.

Tancredi and Liseo [45] describe another Bayesian solution for linear regression where the regression coefficients and the linkage parameters are estimated jointly. A complex Bayesian model is used, including a specification for the prior joint distribution of the linkage variables, the associated recording errors, the pairs match statuses⁶, and the analytical variables (i.e. the covariates and the responses). This model incorporates the following important assumptions. Each file is free of duplicates and contains independent records from independent individuals, with mutually independent linkage variables for each individual. The recorded linkage variables are characterized by errors that are independent across individuals. In the regression model, the response has a normal distribution. The estimation uses a Monte Carlo Markov Chain (MCMC) to sample from the posterior distribution.

Goldstein et al. [42] describe yet another solution using multiple imputations. In their setup, the probabilistic method is used to link a first file of covariates $[\mathbf{x}_i]_i$ with a second file of responses $[y_j]_j$. This linkage assigns the weight w_{ij} to the pair (i, j) . Goldstein et al. propose a Bayesian model, where for each j , a prior distribution (called *data value prior*) is assigned to the actual covariates $\mathbf{x}_{ij}$ based on the observed data $[(\mathbf{x}_i, w_{ij})]_{i=1,2,\dots}$. This prior is a multinomial on the sample of covariates, where the mass on $\mathbf{x}_i$ is proportional to the linkage weight w_{ij} .

In practice, the application of Bayesian methods on large data sets (with millions of records and record pairs) is essentially limited by the required computations.

⁵For example, a one-to-one linkage

⁶Given by a bipartite matching

Chapter 3

Pairwise EEs when linking registers

3.1 Overview

In this chapter, we consider the linkage of two registers of the same finite population. Section 3.2 describes the notation. Section 3.3 lists the assumptions. Section 3.4 proves many results regarding the conditional distribution of the observed responses. These results are key for the proposed estimation procedures described in Section 3.5, including weighted least squares and maximum composite likelihood procedures. Section 3.6 discusses the large sample properties of the proposed estimators. Section 3.7 describes a simulation study.

3.2 Notation

We consider a finite population of N individuals and two related population registers that are linked. Each individual has a set of attributes, which include quasi-identifiers¹, a set of covariates and related responses. The first register (hereafter called A register)

¹A quasi-identifier is a variable that provides some information about an individual but is not unique, i.e. different individuals can have the same quasi-identifier. The first name and birthdate are good examples.

records the quasi-identifiers and the covariates for each individual, while a second register (hereafter called B register) records the quasi-identifiers and the responses for the same individual. The responses and covariates are recorded with no error but the recorded quasi-identifiers have errors in both files. Let $\mathbf{x}_i$ denote the covariates in record i from register A, and $\mathbf{z}_j$ denote the observed responses in record j from register B. Although it is not observed, let $\mathbf{y}_i$ denote the actual responses that are associated with the covariates $\mathbf{x}_i$ from record i , in register A. Let (i, j) denote the record-pair, where records i and j are from registers A and B respectively, and let γ_{ij} denote the comparison vector². Let m_{ij} denote the indicator of the pair match status, where $m_{ij} = 1$ if the pair is matched and $m_{ij} = 0$ if it is unmatched, and let $\mathbf{M} = [m_{ij}]_{1 \leq i, j \leq N}$ denote the match matrix that is a permutation in the current setup. It is also convenient to define the following vectors and matrices:

$$\begin{aligned}\mathbf{z} &= [\mathbf{z}_1^\top \dots \mathbf{z}_N^\top]^\top, \\ \mathbf{y} &= [\mathbf{y}_1^\top \dots \mathbf{y}_N^\top]^\top, \\ \mathbf{X} &= [\mathbf{x}_1^\top \dots \mathbf{x}_N^\top]^\top.\end{aligned}$$

We are interested in estimating the mean $\boldsymbol{\mu} = E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i)]$, or a finite-dimensional parameter $\boldsymbol{\theta}$ such that $E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i; \boldsymbol{\theta})] = \mathbf{0}$, where $\mathbf{g}(\cdot, \cdot; \cdot)$ is some known function.

The model considers H independent and identically distributed (IID) clusters of individuals, which are called blocks, with IID individuals within each block. Block h includes N_h individuals, where $N_h \leq C$ for a constant C regardless of H and $N = N_1 + \dots + N_H$. The same block corresponds to records indexed in two known subsets A_h and B_h (of $\{1, \dots, N\}$) in registers A and B respectively. These subsets contain the same number of records (i.e. $|A_h| = |B_h| = N_h$) and are such that each

²The result of comparing the linkage variables

A record in A_h has a single matching B record in B_h . The collections of subsets $A_1, \dots, A_H$ and $B_1, \dots, B_H$ form two partitions of $\{1, \dots, N\}$. The existence of such known subsets means that each register contains an error-free variable that gives the identity of the corresponding block for each record. This variable provides the basis for a *perfect* blocking criterion, i.e. one met by any matched pair. When generating the record-pairs, the Cartesian product is taken within each block and no pair is formed between records from different blocks. For simplicity, the quasi-identifiers, the covariates and the responses are assumed to have an homogeneous distribution across the blocks. In previous work, Chambers [22] has described a related model, where each block is instead a post-stratum.

3.3 Assumptions

The following assumptions are made.

- A.1 Let $\mathbf{M}_h$ denote the match matrix within block h . It is assumed to have a uniform random permutation independent of the block covariates $[\mathbf{x}_i]_{i \in A_h}$ and the block size N_h .
- A.2 The actual responses $[\mathbf{y}_i]_{i \in A_h}$ are conditionally independent of the match matrix $\mathbf{M}_h$ and the comparison vectors $\{\gamma_{ij}\}_{(i,j) \in A_h \times B_h}$, given the block covariates $[\mathbf{x}_i]_{i \in A_h}$ and the block size N_h .
- A.3 For $(i, j) \in A_h \times B_h$, the conditional match probability $P(m_{i'j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij})$ is the same for all $i' \in A_h - \{i\}$. Also the conditional match probability $P(m_{ij'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij})$ is the same for all $j' \in B_h - \{j\}$.

A.4 The sequence $[(\mathbf{x}_i, \mathbf{y}_i)]_{i \in A_h}$ is IID given the block size N_h , with a common distribution that does not depend on N_h .

A.5 In block h , γ_{ij} is conditionally independent of $[\mathbf{x}_{i''}]_{i'' \in A_h - \{i\}}$ given $\mathbf{x}_i$ and N_h .

3.4 Conditional response distribution

3.4.1 Information from a block

The following theorem is the main result of this section.

Theorem 1 *Suppose that assumptions A.1-A.4 from Section 3.3 apply. Then, for $(i, j) \in A_h \times B_h$, we have*

$$\begin{aligned} E \left[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right] &= q_{ij} E \left[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i \right] + \\ &\quad (1 - q_{ij}) \frac{\sum_{i' \in A_h - \{i\}} E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right]}{N_h - 1}, \end{aligned} \quad (3.1)$$

where $q_{ij} = P(m_{ij} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij})$. Also for any $j' \in B_h - \{j\}$,

$$\begin{aligned} E \left[\mathbf{g}(\mathbf{z}_{j'}; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right] &= \frac{1 - q_{ij}}{N_h - 1} E \left[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i \right] + \\ &\quad \left(1 - \frac{1 - q_{ij}}{N_h - 1} \right) \frac{\sum_{i' \in A_h - \{i\}} E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right]}{N_h - 1}. \end{aligned} \quad (3.2)$$

Proof: To prove Eq. (3.1), first note that

$$\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) = \sum_{i' \in A_h} m_{i'j} \mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i).$$

Hence

$$E \left[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right] = \sum_{i'=1}^N E \left[m_{i'j} \mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right].$$

Using the conditional independence of $[\mathbf{y}_{j'}]_{j' \in B_h}$ and $\left(\mathbf{M}_h, \{\gamma_{i'j'}\}_{(i',j') \in A_h \times B_h} \right)$ given $[\mathbf{x}_{i''}]_{i'' \in A_h}$ and N_h , we have

$$\begin{aligned} & E \left[m_{i'j} \mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right] \\ &= \frac{E \left[I(\gamma_{ij}) m_{i'j} \mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h \right]}{E \left[I(\gamma_{ij}) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h \right]} \\ &= \frac{E \left[I(\gamma_{ij}) m_{i'j} \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h \right] E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h \right]}{E \left[I(\gamma_{ij}) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h \right]} \\ &= P \left(m_{i'j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right) E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h \right] \\ &= P \left(m_{i'j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right) E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right]. \end{aligned}$$

Consequently

$$\begin{aligned} & E \left[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right] \\ &= P \left(m_{ij} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right) E \left[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i \right] + \\ & \quad \sum_{i' \in A_h - \{i\}} P \left(m_{i'j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h \right) E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right]. \end{aligned}$$

To complete the proof of Eq. (3.1), we next show that for $i' \neq i$,

$$P \left(m_{i'j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right) = \frac{1}{N_h - 1} P \left(m_{ij} = 0 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right). \quad (3.3)$$

Indeed, by assumption A.3 from Section 3.3,

$$\begin{aligned}
1 &= \sum_{i' \in A_h} P(m_{i'j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) \\
&= P(m_{ij} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) + \sum_{i' \in A_h - \{i\}} P(m_{i'j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) \\
&= P(m_{ij} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) + (N_h - 1)P(m_{i''j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}),
\end{aligned}$$

where $i'' \neq i$. Thus $P(m_{i''j} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) = P(m_{ij} = 0 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) / (N_h - 1)$ as required.

For Eq. (3.2), consider $j' \in B_h - \{j\}$. Proceeding as before, we have

$$\begin{aligned}
&E[\mathbf{g}(\mathbf{z}_{j'}, \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}] \\
&= P(m_{ij'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i] + \\
&\quad \sum_{i' \in A_h - \{i\}} P(m_{i'j'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}] \\
&= \frac{P(m_{ij} = 0 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij})}{N_h - 1} E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i] + \\
&\quad \sum_{i' \in A_h - \{i\}} P(m_{i'j'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}].
\end{aligned}$$

To complete the proof of Eq. (3.2), we next show that for $i' \neq i$,

$$P(m_{i'j'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) = \frac{1}{N_h - 1} \left(1 - \frac{P(m_{ij} = 0 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij})}{N_h - 1} \right). \quad (3.4)$$

Using again assumption A.3 from Section 3.3,

$$\begin{aligned}
1 &= \sum_{i' \in A_h} P(m_{i'j'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) \\
&= P(m_{ij'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) + \sum_{i' \in A_h - \{i\}} P(m_{i'j'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) \\
&= P(m_{ij'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) + (N_h - 1)P(m_{i''j'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}).
\end{aligned}$$

Therefore,

$$P(m_{i''j'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) = \frac{1}{N_h - 1} (1 - P(m_{ij'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij})).$$

We complete the proof of Eq. (3.4) and then Eq. (3.2) by noting that in the above equation, we have $P(m_{ij'} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) = P(m_{ij} = 0 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) / (N_h - 1)$.

Q.E.D.

The above theorem enables the estimation of the mean $E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i)]$ for any function $\mathbf{g}(\cdot; \cdot)$. Eq. (3.2) is not required for simpler functions that do not involve the covariates $\mathbf{x}_i$, as in a general regression problem, where $E[\mathbf{y}_i \mid \mathbf{x}_i] = \mu(\mathbf{x}_i; \boldsymbol{\theta})$. In this case we let $\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) = \mathbf{y}_i$. However, Eq. (3.2) is the key to estimating the mean $E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i)]$ for a nonlinear function $\mathbf{g}(\cdot; \cdot)$ that can neither be expressed as the product of two separate functions of $\mathbf{x}_i$ and $\mathbf{y}_i$ nor as a finite sum of such products.

In the above theorem, we have

$$\begin{aligned}
q_{ij} &= P(m_{ij} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) \\
&= \left(1 + \left(\frac{1}{N_h} - 1 \right) \frac{P(\gamma_{ij} \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, m_{ij} = 0)}{P(\gamma_{ij} \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, m_{ij} = 1)} \right)^{-1}.
\end{aligned} \tag{3.5}$$

The following corollary gives the conditional second order moment of $\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i)$. It is quite useful for estimation procedures and a direct consequence of the previous theorem.

Corollary 1 *Suppose that assumptions A.1-A.4 from Section 3.3 apply and let Σ_{ij} denote the conditional variance-covariance matrix of $\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i)$ given $[\mathbf{x}_{i''}]_{i'' \in A_h}$, N_h and γ_{ij} , i.e.*

$$\begin{aligned} \Sigma_{ij} = & E \left[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \mathbf{g}(\mathbf{z}_j; \mathbf{x}_i)^\top \middle| [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right] - \\ & E \left[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \middle| [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right] E \left[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \middle| [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij} \right]^\top. \end{aligned} \quad (3.6)$$

Then, for $(i, j) \in A_h \times B_h$, we have

$$\begin{aligned} \Sigma_{ij} = & q_{ij} E \left[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mathbf{g}(\mathbf{y}_i; \mathbf{x}_i)^\top \middle| \mathbf{x}_i \right] + \\ & (1 - q_{ij}) \frac{\sum_{i' \in A_h - \{i\}} E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i)^\top \middle| \mathbf{x}_i, \mathbf{x}_{i'} \right]}{N_h - 1} - \\ & \left(q_{ij} E \left[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \middle| \mathbf{x}_i \right] + (1 - q_{ij}) \frac{\sum_{i' \in A_h - \{i\}} E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \middle| \mathbf{x}_i, \mathbf{x}_{i'} \right]}{N_h - 1} \right) \times \\ & \left(q_{ij} E \left[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \middle| \mathbf{x}_i \right] + (1 - q_{ij}) \frac{\sum_{i' \in A_h - \{i\}} E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \middle| \mathbf{x}_i, \mathbf{x}_{i'} \right]}{N_h - 1} \right)^\top. \end{aligned} \quad (3.7)$$

Proof: Start with Eq. (3.6) and apply Theorem 1 to each term on the right-hand side (RHS). For the first term on the RHS, apply the theorem to $\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \mathbf{g}(\mathbf{z}_j; \mathbf{x}_i)^\top$ instead of $\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i)$.

Q.E.D.

The following corollary gives the conditional distribution of the observed response $\mathbf{z}_j$.

Corollary 2 *Suppose that assumptions A.1-A.4 from Section 3.3 apply. For $(i, j) \in A_h \times B_h$, let O_{ij} denote the event $\{[\mathbf{x}_{i''}]_{i'' \in A_h}\} \cap \{N_h\} \cap \{\gamma_{ij}\}$. Let $f_{y|x}(\cdot|\cdot)$ denote the conditional response distribution, and $f_{ij}(\cdot|\cdot)$ denote the conditional pdf or pmf of $\mathbf{z}_j$ given O_{ij} . We have*

$$f_{ij}(\boldsymbol{\zeta} | O_{ij}) = q_{ij} f_{y|x}(\boldsymbol{\zeta} | \mathbf{x}_i) + (1 - q_{ij}) \frac{\sum_{i' \in A_h - \{i\}} f_{y|x}(\boldsymbol{\zeta} | \mathbf{x}_{i'})}{N_h - 1}, \quad (3.8)$$

where $q_{ij} = P(m_{ij} = 1 | [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij})$.

Proof: Apply Theorem 1 with $g(\mathbf{y}_i; \mathbf{x}_i) = I(\mathbf{y}_i \leq \boldsymbol{\zeta})$ to obtain the conditional CDF of $\mathbf{z}_j$. Then obtain the density (PDF for a continuous response and PMF for a categorical response) as the Radon-Nikodym derivative [46, Section 32, pp. 423] of the CDF.

Q.E.D.

The above result is useful when estimating a parameter by the maximization of a composite likelihood. This is of interest when the conditional response distribution has a parametric form.

3.4.2 Information from a single pair

In this section, we look at the conditional distribution of an observed response vector $\mathbf{z}_j$ given the information of the pair (i, j) . The main result is Theorem 2.

Theorem 2 *Suppose that assumptions A.1-A.5 from Section 3.3 apply. Then, for $(i, j) \in A_h \times B_h$, we have*

$$E[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) | \mathbf{x}_i, \gamma_{ij}] = E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) | \mathbf{x}_i] + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i], \quad (3.9)$$

where $q_{ij} = P(m_{ij} = 1 \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij})$.

Proof: From Theorem 1, we have

$$\begin{aligned} E[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \mid [\mathbf{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}] &= q_{ij} E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i] + \\ &\quad (1 - q_{ij}) \frac{\sum_{i' \in A_h - \{i\}} E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}]}{N_h - 1}. \end{aligned}$$

Therefore,

$$\begin{aligned} E[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \mid \mathbf{x}_i, \gamma_{ij}] &= E[q_{ij} E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i] \mid \mathbf{x}_i, \gamma_{ij}] + \\ &\quad E\left[\frac{1}{N_h - 1} \sum_{i' \in A_h - \{i\}} (1 - q_{ij}) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}] \mid \mathbf{x}_i, \gamma_{ij}\right]. \end{aligned} \quad (3.10)$$

Since $E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i]$ is only a function of $\mathbf{x}_i$, we have

$$E[q_{ij} E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i] \mid \mathbf{x}_i, \gamma_{ij}] = E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mid \mathbf{x}_i] E[q_{ij} \mid \mathbf{x}_i, \gamma_{ij}]. \quad (3.11)$$

We also have

$$\begin{aligned} &E\left[\frac{1}{N_h - 1} \sum_{i' \in A_h - \{i\}} (1 - q_{ij}) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}] \mid \mathbf{x}_i, \gamma_{ij}\right] = \\ &E\left[\frac{1}{N_h - 1} \sum_{i' \in A_h - \{i\}} E[(1 - q_{ij}) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}] \mid \mathbf{x}_i, \gamma_{ij}, N_h] \mid \mathbf{x}_i, \gamma_{ij}\right]. \end{aligned}$$

Using assumptions A.5 and A.4 in Section 3.3, we have

$$\begin{aligned}
& E[(1 - q_{ij}) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | \mathbf{x}_i, \gamma_{ij}, N_h] \\
&= E[(1 - q_{ij}) | \mathbf{x}_i, \gamma_{ij}, N_h] E[E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | \mathbf{x}_i, N_h] \\
&= E[(1 - q_{ij}) | \mathbf{x}_i, \gamma_{ij}, N_h] E[E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | \mathbf{x}_i] \\
&= E[(1 - q_{ij}) | \mathbf{x}_i, \gamma_{ij}, N_h] E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i].
\end{aligned}$$

Hence

$$\begin{aligned}
& E \left[\frac{1}{N_h - 1} \sum_{i' \in A_h - \{i\}} (1 - q_{ij}) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] \middle| \mathbf{x}_i, \gamma_{ij} \right] \\
&= E \left[\left(\frac{1}{N_h - 1} \sum_{i' \in A_h - \{i\}} \right) E[(1 - q_{ij}) | \mathbf{x}_i, \gamma_{ij}, N_h] E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i] \middle| \mathbf{x}_i, \gamma_{ij} \right] \\
&= E[E[(1 - q_{ij}) | \mathbf{x}_i, \gamma_{ij}, N_h] | \mathbf{x}_i, \gamma_{ij}] E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i] \\
&= (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i]. \quad (3.12)
\end{aligned}$$

Conclude by using Eq. (3.11) and Eq. (3.12) in Eq. (3.10).

Q.E.D.

The following corollaries are direct consequences of Theorem 2:

Corollary 3 *Suppose that assumptions A.1-A.5 from Section 3.3 apply. Then, for $(i, j) \in A_h \times B_h$, we have let Σ_{ij} denote the conditional variance-covariance matrix of $\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i)$ given $\mathbf{x}_i$ and γ_{ij} , i.e.*

$$\begin{aligned}
\Sigma_{ij} &= E \left[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) \mathbf{g}(\mathbf{z}_j; \mathbf{x}_i)^\top \middle| \mathbf{x}_i, \gamma_{ij} \right] - \\
&E[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) | \mathbf{x}_i, \gamma_{ij}] E[\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i) | \mathbf{x}_i, \gamma_{ij}]^\top.
\end{aligned}$$

Then, for $(i, j) \in A_h \times B_h$, we have

$$\begin{aligned}
\Sigma_{ij} &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] E \left[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) \mathbf{g}(\mathbf{y}_i; \mathbf{x}_i)^\top \middle| \mathbf{x}_i \right] + \\
&\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E \left[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) \mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i)^\top \middle| \mathbf{x}_i \right] - \\
&\quad (E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) | \mathbf{x}_i] + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i]) \times \\
&\quad (E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] E[\mathbf{g}(\mathbf{y}_i; \mathbf{x}_i) | \mathbf{x}_i] + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[\mathbf{g}(\mathbf{y}_{i'}; \mathbf{x}_i) | \mathbf{x}_i])^\top \\
&\quad , \tag{3.13}
\end{aligned}$$

where $q_{ij} = P(m_{ij} = 1 | \mathbf{x}_i, \gamma_{ij})$.

Proof: The proof is straightforward and similar to that of Corollary 1.

Q.E.D.

Corollary 4 Suppose that assumptions A.1-A.4 from Section 3.3 apply. For $(i, j) \in A_h \times B_h$, let O_{ij} denote the event $\{\mathbf{x}_i\} \cap \{\gamma_{ij}\}$. Define $f_{y|x}(\cdot | \cdot)$ the conditional response distribution, $f_y(\cdot)$ the marginal response distribution and $f_{ij}(\cdot | \cdot)$ the conditional pdf or pmf of z_j given O_{ij} . We have

$$f_{ij}(\boldsymbol{\zeta} | O_{ij}) = E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] f_{y|x}(\boldsymbol{\zeta} | \mathbf{x}_i) + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) f_y(\boldsymbol{\zeta}), \tag{3.14}$$

where $q_{ij} = P(m_{ij} = 1 | \mathbf{x}_i, \gamma_{ij})$.

Proof: The proof is straightforward and similar to that of Corollary 2.

Q.E.D.

3.5 Estimation procedures

Using the results from the previous section, we propose estimation procedures for two kinds of regression problems. In the first problem, we describe a weighted least squares (WLS) procedure to estimate a general parameter $\boldsymbol{\theta}$ such that

$$E[\mathbf{y}_i | \mathbf{x}_i] = \mu(\mathbf{x}_i; \boldsymbol{\theta}), \quad (3.15)$$

where $\mu(\cdot, \cdot)$ is a known function. In the second problem, we describe a maximum composite likelihood estimation (MCLE) to estimate a parameter $\boldsymbol{\theta}$ such that $\mathbf{y}_i | \mathbf{x}_i \sim f_{y|x}(\cdot | \mathbf{x}_i, \boldsymbol{\theta})$, where the conditional distribution $f_{y|x}(\cdot | \mathbf{x}_i, \boldsymbol{\theta})$ has a known parametric form.

In both cases, the estimation of q_{ij} is an important problem. In general this estimation is based on the methods of Section 2.3.2. In that section, we do not consider any dependence of the comparison vector on the covariates but the described are easily extended to settings with low-dimensional categorical covariates, where a separate set of mixture parameters may be estimated for each cross-classification of the covariates. However this solution does not apply with continuous or high-dimensional covariates. In that situation we may first apply dimension reduction techniques, e.g. Principal Components Analysis, to the covariates and then estimate different set of mixture parameters within cells that are based on cross-classifications of the selected principal components. An alternative is using nonparametric procedures including some smoothing. Unfortunately, the existing record-linkage literature has been largely silent regarding this issue.

For simplicity, in the subsequent examples and simulations, we limit ourselves to scenarios where the conditional distributions of the comparison vector γ_{ij} do not depend on the block covariates, i.e. $P(\gamma_{ij} | m_{ij}, [\mathbf{x}_{i'}]_{i' \in A_h}) = P(\gamma_{ij} | m_{ij})$. Let

$\gamma_{ij} = \left(\gamma_{ij}^{(1)}, \dots, \gamma_{ij}^{(K)} \right)$. When the linkage is one-to-one-and-onto, we have $P(m_{ij} = 1) = N_h^{-1}$ if $(i, j) \in A_h \times B_h$. Thus γ_{ij} follows the mixture $N_h^{-1} P(\cdot | m_{ij} = 1; \boldsymbol{\psi}_M) + (1 - N_h^{-1}) P(\cdot | m_{ij} = 0; \boldsymbol{\psi}_U)$, where $\boldsymbol{\psi}_M$ and $\boldsymbol{\psi}_U$ are the underlying parameters for the matched and unmatched distributions respectively. Under the conditional independence assumptions these parameters comprise of the marginal probabilities for each variable. Then the estimation of these parameters may be based on the E-M procedure that is described by Eq. (2.1) and Eq (2.3).

$$\begin{aligned}
q_{ij} &= P(m_{ij} = 1 | [\boldsymbol{x}_{i''}]_{i'' \in A_h}, N_h, \gamma_{ij}) \\
&= P(m_{ij} = 1 | N_h, \gamma_{ij}) \\
&= \frac{P(m_{ij} = 1 | N_h) P(\gamma_{ij} | m_{ij} = 1)}{P(m_{ij} = 1 | N_h) P(\gamma_{ij} | m_{ij} = 1) + P(m_{ij} = 0 | N_h) P(\gamma_{ij} | m_{ij} = 0)} \\
&= \frac{N_h^{-1} P(\gamma_{ij} | m_{ij} = 1)}{N_h^{-1} P(\gamma_{ij} | m_{ij} = 1) + (1 - N_h^{-1}) P(\gamma_{ij} | m_{ij} = 0)} \\
&= \left(1 + (N_h - 1) \frac{P(\gamma_{ij} | m_{ij} = 0)}{P(\gamma_{ij} | m_{ij} = 1)} \right)^{-1}.
\end{aligned}$$

where

$$\begin{aligned}
P(\gamma_{ij} | m_{ij} = 0) &= \prod_{k=1}^K P(\gamma_{ij}^{(k)} | m_{ij} = 0), \\
P(\gamma_{ij} | m_{ij} = 1) &= \prod_{k=1}^K P(\gamma_{ij}^{(k)} | m_{ij} = 1),
\end{aligned}$$

with $P(\gamma_{ij}^{(k)} | m_{ij} = 0)$ and $P(\gamma_{ij}^{(k)} | m_{ij} = 1)$ estimated by the EM procedure mentioned above.

When conditioning on the information from a single pair, we also need to estimate

the conditional mean of q_{ij} , which is here computed as follows.

$$\begin{aligned}
E[q_{ij}|\mathbf{x}_i, \gamma_{ij}] &= E\left[P(m_{ij} = 1|N_h, \gamma_{ij})|\mathbf{x}_i, \gamma_{ij}\right] \\
&= E\left[\left(1 + (N_h - 1) \frac{P(\gamma_{ij}|m_{ij} = 0)}{P(\gamma_{ij}|m_{ij} = 1)}\right)^{-1} \cdot \left|\mathbf{x}_i, \gamma_{ij}\right.\right] \\
&= E\left[\left(1 + (N_h - 1) \frac{P(\gamma_{ij}|m_{ij} = 0)}{P(\gamma_{ij}|m_{ij} = 1)}\right)^{-1} \cdot \left|\gamma_{ij}\right.\right] \\
&= \sum_{N_h} p(N_h) \left(1 + (N_h - 1) \frac{P(\gamma_{ij}|m_{ij} = 0)}{P(\gamma_{ij}|m_{ij} = 1)}\right)^{-1}
\end{aligned}$$

For a constant block size, we have

$$q_{ij} = E[q_{ij}|\mathbf{x}_i, \gamma_{ij}] = \left(1 + (N_h - 1) \frac{P(\gamma_{ij}|m_{ij} = 0)}{P(\gamma_{ij}|m_{ij} = 1)}\right)^{-1}$$

3.5.1 Weighted Least Squares

Consider a general regression problem $E[\mathbf{y}_i|\mathbf{x}_i] = \mu(\mathbf{x}_i; \boldsymbol{\theta})$, for some known function $\mu(., .)$. In this case, let $\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i, \boldsymbol{\theta}) = \mathbf{z}_j$ and let O_{ij} denote the conditioning information (event) for the observed responses $\mathbf{z}_i$. When considering all the covariates in the corresponding block, this event is $\{[\mathbf{x}_{i''}]_{i'' \in A_h}\} \cap \{N_h\} \cap \{\gamma_{ij}\}$. When considering the covariates in the pair (i, j) , this event is $\{\mathbf{x}_i\} \cap \{\gamma_{ij}\}$. Also, let

$$\boldsymbol{\Delta}_{ij}(\boldsymbol{\theta}) = \mathbf{z}_j - E[\mathbf{z}_j|O_{ij}], \quad (3.16)$$

where the conditional expectation $E[\mathbf{z}_j|O_{ij}]$ is given by Theorem 1 or Theorem 2 depending on O_{ij} . When $O_{ij} = \{[\mathbf{x}_{i''}]_{i'' \in A_h}\} \cap \{N_h\} \cap \{\gamma_{ij}\}$, the conditional expectation

$E[\mathbf{z}_j | O_{ij}]$ is given by Theorem 1:

$$E[\mathbf{z}_j | O_{ij}] = q_{ij} \mu(\mathbf{x}_i; \boldsymbol{\theta}) + (1 - q_{ij}) \frac{\sum_{i' \in A_h - \{i\}} \mu(\mathbf{x}_{i'}; \boldsymbol{\theta})}{N_h - 1}. \quad (3.17)$$

When $O_{ij} = \{\mathbf{x}_i\} \cap \{\gamma_{ij}\}$, the conditional expectation $E[\mathbf{z}_j | O_{ij}]$ is given by Theorem 2:

$$E[\mathbf{z}_j | O_{ij}] = E[q_{ij} | O_{ij}] \mu(\mathbf{x}_i; \boldsymbol{\theta}) + (1 - E[q_{ij} | O_{ij}]) E[\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})]. \quad (3.18)$$

We may use the WLS estimator

$$\hat{\boldsymbol{\theta}} = \arg \min_{\boldsymbol{\theta}} \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \boldsymbol{\Delta}_{ij}^\top \mathbf{V}_{ij}^{-1} \boldsymbol{\Delta}_{ij}, \quad (3.19)$$

where $\mathbf{V}_{ij}$ is any symmetric positive definite matrix, and τ_{ij} is any nonnegative and non decreasing function of $E[q_{ij} | O_{ij}]$. The asymptotic variance of the resulting estimator is given in the next section. For a good choice of the matrix $\mathbf{V}_{ij}$, we may refer to the quasi-likelihood (QL) framework [47] that suggests the choice $\mathbf{V}_{ij} = \boldsymbol{\Sigma}_{ij} = E[\boldsymbol{\Delta}_{ij}(\boldsymbol{\theta}_0) \boldsymbol{\Delta}_{ij}(\boldsymbol{\theta}_0)^\top | O_{ij}]$, where $\boldsymbol{\Sigma}_{ij}$ is given by Corollary 1 or Corollary 3 according to O_{ij} . The rationale for using τ_{ij} is to give a greater weight to pairs that have a sufficiently high probability of being matched. For example, τ_{ij} may be a step function based on a conditional match probability threshold. This choice also reduces the computational burden of the estimation procedure. The threshold must be selected with care. Too high a threshold may lead to a poor precision by retaining too few pairs for the estimation. Too low a threshold may also decrease the precision by keeping too many unmatched pairs that contribute little information.

The proposed estimator involves a number of nuisance parameters that must be estimated from the data, including the mixture parameters $\boldsymbol{\psi}$ and other parameters

depending on the chosen estimator. The choice $\mathbf{V}_{ij} = \boldsymbol{\Sigma}_{ij}$ requires the estimation of the variance components, i.e. additional parameters that relate to the variance-covariance matrix $E \left[(\mathbf{y}_i - \mu(\mathbf{x}_i; \boldsymbol{\theta})) (\mathbf{y}_i - \mu(\mathbf{x}_i; \boldsymbol{\theta}))^\top \right]$. It also requires a preliminary estimate of $\boldsymbol{\theta}$. Corollary 1 and Corollary 3 provide the basis for estimating the variance components. In practice, the estimator may be computed in multiple steps using plug-in estimates where required. For example, a multi-step plug-in WLS estimator may be computed as follows:

1. Compute $\hat{\boldsymbol{\psi}}$ the mixture parameters.
2. Compute the LSE $\hat{\boldsymbol{\theta}}^{(1)}$ of $\boldsymbol{\theta}$ using $\hat{\boldsymbol{\psi}}$ as plug-in parameters.
3. Estimate the variance components using $\hat{\boldsymbol{\psi}}$ and $\hat{\boldsymbol{\theta}}^{(1)}$ as plug-in parameters. Also compute the estimator $\hat{\boldsymbol{\Sigma}}_{ij}$ of the conditional variance $\boldsymbol{\Sigma}_{ij}$.
4. Compute the WLS estimator $\hat{\boldsymbol{\theta}}^{(2)}$ based on $\hat{\boldsymbol{\Sigma}}_{ij}$, where the estimated variance components, $\hat{\boldsymbol{\psi}}$ and $\hat{\boldsymbol{\theta}}^{(1)}$ are used as plug-in parameters.

When $O_{ij} = \{\mathbf{x}_i\} \cap \{\gamma_{ij}\}$, we have additional nuisance parameters for the marginal distribution of $\mathbf{x}_i$, which is assumed to have a known parametric form. These parameters are required to compute $E[\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})]$ or $E[\mu(\mathbf{x}_{i'}; \boldsymbol{\theta}) \mu(\mathbf{x}_{i'}; \boldsymbol{\theta})^\top]$ (the latter when choosing $\mathbf{V}_{ij} = \boldsymbol{\Sigma}_{ij}$). An unbiased estimator of this expectation is given by

$$\hat{E}[\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})] = \frac{1}{N} \sum_{i=1}^N \mu(\mathbf{x}_{i'}; \boldsymbol{\theta}) \quad (3.20)$$

Note that the above estimator corresponds to the MLE of $E[\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})]$ if $\mathbf{x}_i$ is a categorical vector.

Linear regression example: Consider the homoschedastic linear model with scalar response y_i such that $E[y_i | \mathbf{x}_i] = \mathbf{x}_i^\top \boldsymbol{\beta}$ and $\text{var}(y_i | \mathbf{x}_i) = \sigma^2$.

We first describe the estimator when using all the covariates from a block. In this case, the nuisance parameters are $\boldsymbol{\psi}$ and σ^2 . Define

$$\Delta_{ij} = z_j - \mathbf{w}_{ij}^\top \boldsymbol{\beta}, \quad (3.21)$$

where

$$\mathbf{w}_{ij} = q_{ij} \mathbf{x}_i + \frac{1 - q_{ij}}{N_h - 1} \sum_{i' \in A_h - \{i\}} \mathbf{x}_{i'}. \quad (3.22)$$

When $\boldsymbol{\psi}$ is given, the LSE of $\boldsymbol{\beta}$ is given by

$$\hat{\boldsymbol{\beta}} = \left(\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \mathbf{w}_{ij} \mathbf{w}_{ij}^\top \right)^{-1} \left(\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \mathbf{w}_{ij} z_j \right). \quad (3.23)$$

When $\boldsymbol{\psi}$ and σ^2 are known, the WLSE of $\boldsymbol{\beta}$ is given by

$$\hat{\boldsymbol{\beta}} = \left(\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \frac{\mathbf{w}_{ij} \mathbf{w}_{ij}^\top}{\sigma_{ij}^2} \right)^{-1} \left(\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \frac{\mathbf{w}_{ij} z_j}{\sigma_{ij}^2} \right), \quad (3.24)$$

where

$$\begin{aligned} \sigma_{ij}^2 &= q_{ij} \left(\sigma^2 + (\mathbf{x}_i^\top \boldsymbol{\beta})^2 \right) + \frac{1 - q_{ij}}{N_h - 1} \sum_{i' \in A_h - \{i\}} \left(\sigma^2 + (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2 \right) - (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2 \\ &= \sigma^2 + q_{ij} (\mathbf{x}_i^\top \boldsymbol{\beta})^2 + \frac{1 - q_{ij}}{N_h - 1} \sum_{i' \in A_h - \{i\}} (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2 - (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2. \end{aligned} \quad (3.25)$$

When $\boldsymbol{\psi}$ and $\boldsymbol{\beta}$ are known, a consistent estimator of σ^2 is

$$\hat{\sigma}^2 = \frac{\max \left(0, \left[\sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \tau_{ij} \left((z_j - \mathbf{w}_{ij}^\top \boldsymbol{\beta})^2 - q_{ij} (\mathbf{x}_i^\top \boldsymbol{\beta})^2 - \frac{1 - q_{ij}}{N_h - 1} \sum_{i' \in A_h - \{i\}} (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2 + (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2 \right) \right] \right)}{\sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \tau_{ij}}. \quad (3.26)$$

When $\boldsymbol{\psi}$ is known but σ^2 is unknown, the following procedure may be used for $\boldsymbol{\beta}$. First, the LSE of $\boldsymbol{\beta}$ may be plugged into Eq. (3.26) to estimate σ^2 . In turn these two estimators may be plugged into Eq. (3.25) to estimate σ_{ij}^2 . Finally this latter estimator may be plugged into Eq. (3.24) to produce the multi-step WLSE of $\boldsymbol{\beta}$. When $\boldsymbol{\psi}$ is also unknown, the procedure is modified by using a consistent estimator (e.g., a maximum composite likelihood estimator as described in Section 2.3.2) of this parameter in each step of the procedure.

We now describe estimators that only use the information from a single record pair, say (i, j) . Now, the nuisance parameters include $\boldsymbol{\psi}$, σ^2 , $E[\mathbf{x}_{i'}]$ and $E[\mathbf{x}_{i'} \mathbf{x}_{i'}^\top]$. We still define Δ_{ij} according to Eq. (3.21), with

$$\mathbf{w}_{ij} = E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] \mathbf{x}_i + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[\mathbf{x}_{i'}]. \quad (3.27)$$

When all the nuisance parameters are given, the LSE and WLSE of $\boldsymbol{\beta}$ are given by Eqs. (3.23) and (3.24), respectively. Now τ_{ij} is a nondecreasing function of $E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]$ and σ_{ij}^2 is computed as

$$\sigma_{ij}^2 = \sigma^2 + E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] (\mathbf{x}_i^\top \boldsymbol{\beta})^2 + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[(\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2] - (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2. \quad (3.28)$$

When $\boldsymbol{\psi}$, $\boldsymbol{\beta}$, $E[\mathbf{x}_{i'}]$ and $E[\mathbf{x}_{i'}\mathbf{x}_{i'}^\top]$ are known, a consistent estimator of σ^2 is

$$\hat{\sigma}^2 = \frac{\max \left(0, \left[\sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \tau_{ij} \left((z_j - \mathbf{w}_{ij}^\top \boldsymbol{\beta})^2 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] (\mathbf{x}_i^\top \boldsymbol{\beta})^2 - \right. \right. \right. \\ \left. \left. \left. (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[(\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2] + (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2 \right) \right] \right)}{\sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \tau_{ij}}. \quad (3.29)$$

When the nuisance parameters $\boldsymbol{\psi}$, $E[\mathbf{x}_{i'}]$ and $E[\mathbf{x}_{i'}\mathbf{x}_{i'}^\top]$ are given, the following multi-step procedure may be used for $\boldsymbol{\beta}$. As before, first compute the LSE of $\boldsymbol{\beta}$. Then plug it into Eq. (3.29) to estimate σ^2 and then into Eq. (3.28) (along with the LSE) to compute $\hat{\sigma}_{ij}^2$. Finally, compute the WLSE according to Eq. (3.24) (where $\mathbf{w}_{ij}$ is given by Eq. (3.27)), where $\hat{\sigma}_{ij}^2$ is plugged in. In practice, modify each step of the procedure by plugging in a consistent estimator $\hat{\boldsymbol{\psi}}$ of the mixture parameters as well as the following unbiased estimators of $E[\mathbf{x}_{i'}]$ and $E[\mathbf{x}_{i'}\mathbf{x}_{i'}^\top]$:

$$\hat{E}[\mathbf{x}_{i'}] = \frac{1}{N} \sum_{i'=1}^N \mathbf{x}_{i'}, \quad (3.30)$$

$$\hat{E}[\mathbf{x}_{i'}\mathbf{x}_{i'}^\top] = \frac{1}{N} \sum_{i'=1}^N \mathbf{x}_{i'}\mathbf{x}_{i'}^\top. \quad (3.31)$$

Logistic regression example: Consider a dichotomous response y_i such that $E[y_i | \mathbf{x}_i] = \mu_i = e^{\mathbf{x}_i^\top \boldsymbol{\beta}} / (1 + e^{\mathbf{x}_i^\top \boldsymbol{\beta}})$. For the estimation procedure based on Theorem 1, $E[z_j | O_{ij}]$ is

$$E[z_j | O_{ij}] = q_{ij} \mu_i + \frac{1 - q_{ij}}{N_h - 1} \sum_{i' \in A_h - \{i\}} \mu_{i'} \quad (3.32)$$

The only nuisance parameter is $\boldsymbol{\psi}$. When it is given, the LSE of $\boldsymbol{\beta}$ is

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta}} \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \Delta_{ij}^2. \quad (3.33)$$

As for the WLSE, it is

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta}} \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \frac{\Delta_{ij}^2}{\sigma_{ij}^2}, \quad (3.34)$$

where

$$\sigma_{ij}^2 = E[z_j | O_{ij}] (1 - E[z_j | O_{ij}]) \quad (3.35)$$

To compute the WLSE, we first compute the LSE of $\boldsymbol{\beta}$ using a consistent estimator $\hat{\boldsymbol{\psi}}$ of $\boldsymbol{\psi}$ in Eq. (3.33). We next plug in these two estimators to estimate σ_{ij}^2 and $\boldsymbol{\beta}$ in Eq. (3.35) and Eq. (3.34), respectively.

For the estimation procedure based on Theorem 2, $E[z_j | O_{ij}]$ is

$$E[z_j | O_{ij}] = E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] \mu_i + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[\mu_{i'}]. \quad (3.36)$$

Now, the nuisance parameters include $\boldsymbol{\psi}$ and additional parameters from the marginal distribution of $\mathbf{x}_i$, which we assume to be categorical in this example. Then the nuisance parameters include $\boldsymbol{\psi}$ and the PMF of the covariates. This PMF may be estimated using the empirical distribution of the covariates in file A. Then this

estimated PMF may be used to compute the estimators

$$\widehat{E}[\mu_{i'}] = \frac{1}{N} \sum_{i'=1}^N \mu_{i'}, \quad (3.37)$$

$$\widehat{E}[\mu_{i'}^2] = \frac{1}{N} \sum_{i'=1}^N \mu_{i'}^2, \quad (3.38)$$

for each β . A simple multi-step procedure uses these estimators (and $\widehat{\psi}$) to first compute the LSE and then the WLSE according to Eq. 3.34, where σ_{ij}^2 is based on the LSE.

3.5.2 Maximum composite likelihood

A composite likelihood is the product of simpler component likelihoods for selected subsets of the data [48]. It is called *marginal* or *conditional* according to whether its components are marginal or conditional likelihoods, respectively. In this framework, the estimation is based on the maximization of the composite likelihood to get a maximum composite likelihood estimator (MCLE). This estimator is typically the solution of the unbiased estimating equation, where all partial derivatives of the composite likelihood are set to zero. The corresponding large-sample theory borrows from previous work on estimating equations and misspecified models, including results that naturally extend those of the maximum likelihood framework. These results include the asymptotic normal distribution of the MCLE, the asymptotic chi-square mixture distribution for the composite likelihood ratio statistic. Composite likelihoods were initially used to deal with situations where the joint likelihood is intractable. However they provide further benefits such as greatly reduced risks of model misspecification, and simpler and more stable numerical procedures, which include EM procedures for scenarios with missing or incomplete data. In our setting, the composite likelihood

method is a natural choice when we have a parametric conditional distribution for the actual responses $(\mathbf{y}_i)$ given the covariates $(\mathbf{x}_i)$ for $i = 1, \dots, N$. Then a marginal composite likelihood may be defined as a product of marginal conditional likelihoods over selected pairs, where the component for pair (i, j) is given by Corollary 2 or Corollary 4 depending on the conditioning information O_{ij} . The proposed composite log-likelihood estimator is the solution of the maximization problem

$$\hat{\boldsymbol{\theta}} = \arg \max_{\boldsymbol{\theta}} \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \log f_{ij}(\mathbf{z}_j | O_{ij}), \quad (3.39)$$

where $O_{ij} = \{[\mathbf{x}_{i''}]_{i'' \in A_h}\} \cap \{N_h\} \cap \{\gamma_{ij}\}$ or $O_{ij} = \{\mathbf{x}_i\} \cap \{\gamma_{ij}\}$ and τ_{ij} is any nonnegative and non decreasing function of $E[q_{ij} | O_{ij}]$. As with the WLS estimator, τ_{ij} may be a step function of $E[q_{ij} | O_{ij}]$ based on a threshold.

Survival model example: For individual i in the finite population, we have a set of covariates $\mathbf{x}_i$, a right-censored survival time $t_i \leq T$ where T is the known duration of the follow-up. A parametric PHM is assumed with a constant hazard, i.e. exponential survival times. For each individual, file B records the survival times, while file A records the covariates $\mathbf{x}_i$. Let $f_{t|x}(\cdot | \cdot)$ denote the conditional distribution of the censored survival time given the covariates. It is

$$f_{t|x}(t_i | \mathbf{x}_i) = I(t_i < T) e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \exp\left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} t_i\right) + I(t_i = T) \exp\left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T\right). \quad (3.40)$$

Let us first consider the estimation procedure based on Corollary 2, where all the

block covariates are used. Then $f_{ij}(\cdot | O_{ij})$ is

$$\begin{aligned}
 f_{ij}(z | O_{ij}; \boldsymbol{\beta}) &= q_{ij} f_{t|x}(z | \mathbf{x}_i) + \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h - \{i\}} f_{t|x}(z | \mathbf{x}_{i'}) \\
 &= q_{ij} \left(I(z < T) e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \exp(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}}) + I(z = T) \exp(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T) \right) + \\
 &\quad \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h - \{i\}} \left(I(z < T) e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \exp(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z) + \right. \\
 &\quad \left. I(z = T) \exp(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T) \right). \tag{3.41}
 \end{aligned}$$

Then the MCLE is

$$\hat{\boldsymbol{\beta}} = \arg \max_{\boldsymbol{\beta}} \underbrace{\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \log f_{ij}(z_j | O_{ij}; \boldsymbol{\beta})}_{\ell(\boldsymbol{\beta})}, \tag{3.42}$$

where $f_{ij}(\cdot | O_{ij}; \boldsymbol{\beta})$ is according to Eq. (3.41). The MCLE is a stationary point of the composite log-likelihood $\ell(\boldsymbol{\beta})$, i.e. $\hat{\boldsymbol{\beta}}$ satisfies the equation

$$\frac{\partial \ell}{\partial \boldsymbol{\beta}^\top} = \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \frac{\partial}{\partial \boldsymbol{\beta}^\top} \log f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) = \mathbf{0}, \tag{3.43}$$

where

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\beta}^\top} \log f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) &= \frac{1}{f_{ij}(z_j | O_{ij}; \boldsymbol{\beta})} \frac{\partial}{\partial \boldsymbol{\beta}^\top} f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) \\
&= \frac{q_{ij}}{f_{ij}(z_j | O_{ij}; \boldsymbol{\beta})} \left[\frac{I(z_j < T) (1 - z_j e^{\mathbf{x}_i^\top \boldsymbol{\beta}})}{\exp(-\mathbf{x}_i^\top \boldsymbol{\beta} + e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z_j)} - \right. \\
&\quad \left. \frac{I(z_j = T) T e^{\mathbf{x}_i^\top \boldsymbol{\beta}}}{\exp(e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T)} \right] \mathbf{x}_i^\top + \\
&\quad \frac{(1 - q_{ij})}{f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) (N_h - 1)} \sum_{i' \in A_h - \{i\}} \left[\frac{I(z_j < T) (1 - z_j e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}})}{\exp(-\mathbf{x}_{i'}^\top \boldsymbol{\beta} + e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z_j)} - \right. \\
&\quad \left. \frac{I(z_j = T) T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}}}{\exp(e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T)} \right] \mathbf{x}_{i'}^\top. \tag{3.44}
\end{aligned}$$

The solution may be computed numerically using an iterative Newton-Raphson procedure that operates as follows. Let $\widehat{\boldsymbol{\beta}}^{(s)}$ denote the estimate in iteration s . Then $\widehat{\boldsymbol{\beta}}^{(s+1)}$ is

$$\widehat{\boldsymbol{\beta}}^{(s+1)} = \widehat{\boldsymbol{\beta}}^{(s)} - \left(\frac{\partial^2 \ell}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} \Big|_{\widehat{\boldsymbol{\beta}}^{(s)}} \right)^{-1} \left(\frac{\partial \ell}{\partial \boldsymbol{\beta}^\top} \Big|_{\widehat{\boldsymbol{\beta}}^{(s)}} \right). \tag{3.45}$$

The second-order derivative of the composite log-likelihood is computed as

$$\frac{\partial^2 \ell}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} = \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} \log f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) = \mathbf{0}, \tag{3.46}$$

where

$$\begin{aligned} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} \log f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) &= \frac{1}{f_{ij}(z_j | O_{ij}; \boldsymbol{\beta})} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) - \\ &\frac{1}{f_{ij}(z_j | O_{ij}; \boldsymbol{\beta})^2} \left(\frac{\partial}{\partial \boldsymbol{\beta}^\top} f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) \right) \times \\ &\left(\frac{\partial}{\partial \boldsymbol{\beta}^\top} f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) \right)^\top, \end{aligned} \quad (3.47)$$

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\beta}^\top} f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) &= q_{ij} \left[\frac{I(z_j < T) (1 - z_j e^{\mathbf{x}_i^\top \boldsymbol{\beta}})}{\exp(-\mathbf{x}_i^\top \boldsymbol{\beta} + e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z_j)} - \right. \\ &\left. \frac{I(z_j = T) T e^{\mathbf{x}_i^\top \boldsymbol{\beta}}}{\exp(e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T)} \right] \mathbf{x}_i^\top + \\ &\frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h - \{i\}} \left[\frac{I(z_j < T) (1 - z_j e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}})}{\exp(-\mathbf{x}_{i'}^\top \boldsymbol{\beta} + e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z_j)} - \right. \\ &\left. \frac{I(z_j = T) T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}}}{\exp(e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T)} \right] \mathbf{x}_{i'}^\top, \end{aligned} \quad (3.48)$$

and

$$\begin{aligned}
\frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) &= q_{ij} \left[\frac{I(z_j < T) \left(\left(1 - z_j e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right)^2 - z_j e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right)}{\exp(-\mathbf{x}_i^\top \boldsymbol{\beta} + e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z_j)} - \right. \\
&\quad \left. \frac{I(z_j = T) \left(T e^{\mathbf{x}_i^\top \boldsymbol{\beta}} - \left(T e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right)^2 \right)}{\exp(e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T)} \right] \mathbf{x}_i \mathbf{x}_i^\top + \\
&\quad \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h - \{i\}} \left[\frac{I(z_j < T) \left(\left(1 - z_j e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)^2 - z_j e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)}{\exp(-\mathbf{x}_{i'}^\top \boldsymbol{\beta} + e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z_j)} - \right. \\
&\quad \left. \frac{I(z_j = T) \left(T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} - \left(T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)^2 \right)}{\exp(e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T)} \right] \mathbf{x}_{i'} \mathbf{x}_{i'}^\top. \tag{3.49}
\end{aligned}$$

For the estimation procedure based on Corollary 4 (where only all the pair covariates are used), Eqs. (3.42), (3.43), (3.44), (3.46) and (3.47) still apply. However, the

following changes are required:

$$\begin{aligned}
f_{ij}(z | O_{ij}; \boldsymbol{\beta}) &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] f_{t|x}(z | \mathbf{x}_i) + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E[f_{t|x}(z | \mathbf{x}_{i'})] \\
&= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] \left(I(z < T) e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \exp(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z) + \right. \\
&\quad \left. I(z = T) \exp(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T) \right) + \\
&\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E \left[I(z < T) e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \exp(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z) + \right. \\
&\quad \left. I(z = T) \exp(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T) \right], \tag{3.50}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\beta}^\top} f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] \left[\frac{I(z_j < T) (1 - z_j e^{\mathbf{x}_i^\top \boldsymbol{\beta}})}{\exp(-\mathbf{x}_i^\top \boldsymbol{\beta} + e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z_j)} - \right. \\
&\quad \left. \frac{I(z_j = T) T e^{\mathbf{x}_i^\top \boldsymbol{\beta}}}{\exp(e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T)} \right] \mathbf{x}_i^\top + \\
&\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) E \left[\left(\frac{I(z_j < T) (1 - z_j e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}})}{\exp(-\mathbf{x}_{i'}^\top \boldsymbol{\beta} + e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z_j)} - \right. \right. \\
&\quad \left. \left. \frac{I(z_j = T) T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}}}{\exp(e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T)} \right) \mathbf{x}_{i'}^\top \right], \tag{3.51}
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} f_{ij}(z_j | O_{ij}; \boldsymbol{\beta}) &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}] \left(\frac{I(z_j < T) \left((1 - z_j e^{\mathbf{x}_i^\top \boldsymbol{\beta}})^2 - z_j e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right)}{\exp(-\mathbf{x}_i^\top \boldsymbol{\beta} + e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z_j)} - \right. \\
&\quad \left. \frac{I(z_j = T) \left(T e^{\mathbf{x}_i^\top \boldsymbol{\beta}} - (T e^{\mathbf{x}_i^\top \boldsymbol{\beta}})^2 \right)}{\exp(e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T)} \right) \mathbf{x}_i \mathbf{x}_i^\top + \\
&\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}]) \times \\
&\quad E \left[\left(\frac{I(z_j < T) \left((1 - z_j e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}})^2 - z_j e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)}{\exp(-\mathbf{x}_{i'}^\top \boldsymbol{\beta} + e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z_j)} - \right. \right. \\
&\quad \left. \left. \frac{I(z_j = T) \left(T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} - (T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}})^2 \right)}{\exp(e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T)} \right) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top \right]. \quad (3.52)
\end{aligned}$$

3.6 Large sample theory

We next discuss the consistency and asymptotic normality of the proposed estimators, when $H \rightarrow \infty$. These estimators are essentially z-estimators, which are consistent and asymptotically normal under general conditions given by Van der Vaart [49]. For the consistency of $\widehat{\boldsymbol{\theta}}$, we can apply the following theorem by Van der Vaart [49, pp. 46, Theorem 5.9].

Theorem 3 (Van der Vaart [49]) *Let $\phi_n(\cdot)$ be a random vector-valued function and let ϕ_∞ be a fixed vector-valued function of $\boldsymbol{\theta}$ such that $\|\phi_\infty(\boldsymbol{\theta}_0)\| = 0$ and for*

every $\epsilon > 0$

$$\sup_{\boldsymbol{\theta} \in \Theta} \|\boldsymbol{\phi}_n(\boldsymbol{\theta}) - \boldsymbol{\phi}_\infty(\boldsymbol{\theta})\| \xrightarrow{p} 0 \quad (3.53)$$

$$\inf_{\boldsymbol{\theta}: d(\boldsymbol{\theta}, \boldsymbol{\theta}_0) \geq \epsilon} \|\boldsymbol{\phi}_\infty(\boldsymbol{\theta})\| > 0, \quad (3.54)$$

where $d(.,.)$ is some distance. Then any sequence of estimators $\hat{\boldsymbol{\theta}}_n$ such that $\boldsymbol{\phi}_n(\hat{\boldsymbol{\theta}}_n) = o_p(1)$ converges in probability to $\boldsymbol{\theta}_0$.

According to Van der Vaart [49, pp. 46], Eq. (3.53) is satisfied if the following conditions are met.

1. The parameter space Θ is compact.
2. $\boldsymbol{\phi}_\infty(\boldsymbol{\theta}) = E[\boldsymbol{\phi}(\mathbf{x}_i; \boldsymbol{\theta})]$ and $\boldsymbol{\phi}_n(\cdot)$ is of the form

$$\boldsymbol{\phi}_n(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n \boldsymbol{\phi}(\mathbf{x}_i; \boldsymbol{\theta}),$$

for some function $\boldsymbol{\phi}(\cdot; \cdot)$ and IID sample $\mathbf{x}_1, \dots, \mathbf{x}_n$ (unrelated to our previously defined covariates).

3. The functions $\boldsymbol{\theta} \mapsto \boldsymbol{\phi}(\mathbf{x}; \boldsymbol{\theta})$ are continuous for every $\mathbf{x}$ and dominated by a function of $\mathbf{x}$ that is integrable.

When $\boldsymbol{\phi}_n(\cdot)$ is of the form given by the second condition, Eq. (3.53) means that the family of functions $\{\boldsymbol{\phi}(\cdot; \boldsymbol{\theta}), \boldsymbol{\theta} \in \Theta\}$ is Glivenko-Cantelli³. This is the case if the third condition is satisfied. When all the nuisance parameters⁴ are given, we can

³Let $X_1, \dots, X_n$ be a random sample from a probability distribution P on a measurable space $(\mathcal{X}, \mathcal{A})$. Let $Pf = \int f dP$ denote the expectation of f under P . A family $\mathcal{F}$ of measurable functions $f: \mathcal{X} \mapsto R$ is called *P-Glivenko-Cantelli* if $\sup_{f \in \mathcal{F}} \left| \frac{1}{n} \sum_{i=1}^n f(X_i) - Pf \right| \xrightarrow{as*} 0$ [49, Section 19.2, pp. 269].

⁴For example the mixture parameters, variance components such as σ^2 in linear regression, parameters associated with the marginal distribution of $\mathbf{x}_i$, when conditioning on the information of a single pair.

apply the above theorem by defining $\phi(., \boldsymbol{\theta})$ to be a function of a block as follows. Let O_{ij} denote the event information for Δ_{ij} . It is either $\{N_h\} \cap \{[\mathbf{x}_{i''}]_{i'' \in A_h}\} \cap \{\gamma_{ij}\}$, or simply $\{\mathbf{x}_i\} \cap \{\gamma_{ij}\}$. For the proposed WLS estimators, let

$$\phi\left([\mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}]_{(i,j) \in A_h \times B_h}, N_h, \boldsymbol{\theta}\right) = \sum_{(i,j) \in A_h \times B_h} \tau_{ij} E \left[\left(\frac{\partial \Delta_{ij}}{\partial \boldsymbol{\theta}^\top} \bigg|_{\boldsymbol{\theta}_0} \right) \bigg| O_{ij} \right]^\top \boldsymbol{\Sigma}_{ij}^{-1} \Delta_{ij}, \quad (3.55)$$

$$\phi_\infty(\boldsymbol{\theta}) = E \left[\phi\left([\mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}]_{(i,j) \in A_h \times B_h}, N_h, \boldsymbol{\theta}\right) \right], \quad (3.56)$$

and assume that the conditions of Theorem 3 are met. For the MCLE, replace Eq. (3.55) by the equation

$$\phi\left([\mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}]_{(i,j) \in A_h \times B_h}, N_h, \boldsymbol{\theta}\right) = \sum_{(i,j) \in A_h \times B_h} \tau_{ij} \frac{\partial \log f_{ij}(\mathbf{z}_j | O_{ij})}{\partial \boldsymbol{\theta}^\top}. \quad (3.57)$$

To study the asymptotic normality of $\hat{\boldsymbol{\theta}}$ we can apply the following theorem by Van der Vaart [49, Theorem 5.21, pp. 52], where $\phi(., \boldsymbol{\theta})$ is the function given by Eq. (3.55) or Eq. (3.57) above.

Theorem 4 (Van der Vaart [49]) *For each $\boldsymbol{\theta}$ in an open subset of Euclidian space, let $\mathbf{x} \mapsto \phi(\mathbf{x}; \boldsymbol{\theta})$ be a measurable vector-valued function such that, for every $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ in a neighborhood of $\boldsymbol{\theta}_0$ and a measurable function $\dot{\phi}$ such that $E \left[\left(\dot{\phi}(\mathbf{x}) \right)^2 \right] < \infty$, we have*

$$\|\phi(\mathbf{x}; \boldsymbol{\theta}_1) - \phi(\mathbf{x}; \boldsymbol{\theta}_2)\| \leq \dot{\phi}(\mathbf{x}) \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|. \quad (3.58)$$

Assume that $E[\|\phi(\mathbf{x}; \boldsymbol{\theta}_0)\|^2] < \infty$, $E[\phi(\mathbf{x}; \boldsymbol{\theta}_0)] = \mathbf{0}$, and that the map $\boldsymbol{\theta} \mapsto E[\phi(\mathbf{x}; \boldsymbol{\theta})]$ is differentiable at $\boldsymbol{\theta}_0$, with nonsingular derivative matrix $\mathbf{V}$. If

$$\frac{1}{n} \sum_{i=1}^n \phi(\mathbf{x}_i; \hat{\boldsymbol{\theta}}_n) = o_p(n^{-1/2}) \quad (3.59)$$

and $\hat{\boldsymbol{\theta}}_n \xrightarrow{p} \boldsymbol{\theta}_0$, then

$$\sqrt{n} \left(\hat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}_0 \right) = -\mathbf{V}^{-1} \frac{1}{\sqrt{n}} \sum_{i=1}^n \boldsymbol{\phi}(\mathbf{x}_i; \boldsymbol{\theta}_0) + o_p(1). \quad (3.60)$$

In particular, the sequence $\sqrt{n} \left(\hat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}_0 \right)$ is asymptotically normal with mean zero and covariance matrix $\mathbf{V}^{-1} E \left[\boldsymbol{\phi}(\mathbf{x}; \boldsymbol{\theta}_0) \boldsymbol{\phi}(\mathbf{x}; \boldsymbol{\theta}_0)^\top \right] (\mathbf{V}^{-1})^\top$.

Under the general conditions of Theorem 4, we have the asymptotic normality of the proposed WLS and MCLE. For the WLS, we have

$$\mathbf{V}(\boldsymbol{\theta}_0) = \sum_{h=1}^H E \left[\sum_{(i,j) \in A_h \times B_h} \tau_{ij} E \left[\left(\frac{\partial \Delta_{ij}}{\partial \boldsymbol{\theta}^\top} \middle|_{\boldsymbol{\theta}_0} \right) O_{ij} \right]^\top \boldsymbol{\Sigma}_{ij}^{-1} E \left[\left(\frac{\partial \Delta_{ij}}{\partial \boldsymbol{\theta}^\top} \middle|_{\boldsymbol{\theta}_0} \right) O_{ij} \right] \right]. \quad (3.61)$$

For the MCLE, we have

$$\mathbf{V}(\boldsymbol{\theta}_0) = \sum_{h=1}^H E \left[\sum_{(i,j) \in A_h \times B_h} \tau_{ij} \left(\frac{\partial^2 \log f_{ij}(\mathbf{z}_j | O_{ij})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\top} \middle|_{\boldsymbol{\theta}_0} \right) \right]. \quad (3.62)$$

When each nuisance parameter is not given but estimated, the above theorems still apply if the corresponding estimator is the solution of an unbiased estimating function (UEF), which is a sum of IID contributions over the different blocks⁵. In this case, the above definitions of $\boldsymbol{\phi}(\cdot, \boldsymbol{\theta})$ (i.e. Eq. (3.55) or Eq. (3.57)) are easily extended to include each corresponding UEF, with a corresponding extension of the parameter space.

⁵This is the case for the mixture parameters

For example, when the mixture parameters $\boldsymbol{\psi}$ are estimated⁶, we may instead define $\phi(\cdot, \boldsymbol{\theta}, \boldsymbol{\psi})$ for the WLS by

$$\begin{aligned} & \phi \left([(\mathbf{x}_i, \mathbf{z}_j, \gamma_{ij})]_{(i,j) \in A_h \times B_h}, N_h, \boldsymbol{\theta}, \boldsymbol{\psi} \right) \\ = & \sum_{(i,j) \in A_h \times B_h} \left[\left(\frac{\partial \log P(\gamma_{ij}; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}^\top} \right)^\top \left(\tau_{ij} E \left[\left(\frac{\partial \Delta_{ij}}{\partial \boldsymbol{\theta}^\top} \middle|_{\boldsymbol{\theta}_0} \right) \middle| O_{ij} \right]^\top \boldsymbol{\Sigma}_{ij}^{-1} \Delta_{ij} \right)^\top \right]^\top. \end{aligned} \quad (3.63)$$

For the MCLE, we can use

$$\begin{aligned} & \phi \left([(\mathbf{x}_i, \mathbf{z}_j, \gamma_{ij})]_{(i,j) \in A_h \times B_h}, N_h, \boldsymbol{\theta}, \boldsymbol{\psi} \right) \\ = & \sum_{(i,j) \in A_h \times B_h} \left[\left(\frac{\partial \log P(\gamma_{ij}; \boldsymbol{\psi})}{\partial \boldsymbol{\psi}^\top} \right)^\top \left(\tau_{ij} \frac{\partial \log f_{ij}(\mathbf{z}_j | O_{ij})}{\partial \boldsymbol{\theta}^\top} \right)^\top \right]^\top. \end{aligned} \quad (3.64)$$

3.7 Simulation study

A Monte-Carlo simulation study is conducted for a linear model, a logistic model and a parametric propotional hazards model (PHM). The following paragraphs are organized as follows. Section 3.7.1 describes the general setup. Section 3.7.2 presents the results for the linear model. Section 3.7.3 presents the results for the logistic model. Section 3.7.4 presents the results for the survival model. Section 3.7.5 presents the conclusions.

3.7.1 General setup

The Monte-Carlo simulations involve 100 repetitions for each model (linear, logistic or exponential proportional hazards model), where each repetition includes the following

⁶Assuming that all the other nuisance parameters are known.

three steps in sequence. In the first step, the finite population is generated, including $H = 128$ blocks with a uniform size of $N_h = 2$ or $N_h = 8$, IID individuals within each block and a homogeneous distribution of the individuals across the blocks. For each individual, the corresponding attributes are generated, including $K = 8$ independent Bernoulli linkage variables with probability 0.5, as well as a scalar covariate x and a response y based on the selected model. These attributes are recorded in two registers A and B as follows. In register A, the linkage variables and the covariates are recorded. In register B, the linkage variables and the responses are recorded. The recorded response and covariates are error-free but the linkage variables are recorded with random errors. For each register, each individual and each linkage variable, an independent error is made with probability 0.1. Note that this error is also independent of the individual's covariates and response.

In the second step, the two registers are linked using the probabilistic method. This includes the creation of the potential pairs based on the Cartesian product within each block, the comparison of the recorded linkage variables based on exact agreement and the estimation of the mixture parameters with an EM procedure under the assumption of conditional independence. Note that this assumption is valid given the above data generation mechanism. The second step also includes the estimation of the conditional match probability and the linkage of each pair where the conditional match probability is no less than 0.9. This linkage decision is later used to compute naive estimators of the regression coefficients that ignore the linkage errors.

In the third step, different estimators of the regression coefficients are computed, including a naive estimator that ignores the linkage errors, a complete data estimator that uses the actual match status of pairs, and two estimators using the proposed pairwise method. The performance of the estimators is measured in terms of relative bias, variance and mean squared error (MSE). The following sections give further

details according to the model. The corresponding R code is provided in the appendix, in Section [B.1](#).

3.7.2 Linear model

A homoscedastic linear model is considered, where the regression coefficients are $[\beta_0 \ \beta_1] = [0.5 \ 1]$. For this model, five estimators are evaluated including the naive estimator, the complete data estimator, the LL estimator⁷ and two WLS pairwise estimators according to the methodology of Section [3.5.1](#). The naive estimator is the solution of the biased EE

$$\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} l_{ij} \mathbf{x}_i (z_j - \mathbf{x}_i^\top \boldsymbol{\beta}) = \mathbf{0}, \quad (3.65)$$

where $\mathbf{x}_i = [1 \ x_i]^\top$ and $l_{ij} = I(q_{ij} \geq q_0)$, where $q_{ij} = P(m_{ij} = 1 | \gamma_{ij})$ and $q_0 = 0.9$.

The complete data estimator is the solution of the unbiased EE

$$\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} m_{ij} \mathbf{x}_i (z_j - \mathbf{x}_i^\top \boldsymbol{\beta}) = \mathbf{0}. \quad (3.66)$$

The first pairwise estimator (later called PW1) is based on the conditional distribution of z_j given γ_{ij} and all the covariates observed in the corresponding block. It is the WLS estimator where $v_{ij} = \sigma_{ij}^2$ and $\tau_{ij} = I(q_{ij} \geq q_0)$. The variance σ_{ij}^2 is estimated by using a preliminary estimate of $\boldsymbol{\beta}$ and an estimate of the variance σ^2 based on this preliminary estimate. The preliminary estimate of $\boldsymbol{\beta}$ is based on $v_{ij} = 1$ with the same choice for τ_{ij} . The second pairwise estimator (later called PW2) is based on the conditional distribution of z_j given γ_{ij} and $\mathbf{x}_i$. For this second PW estimator, the variance σ_{ij}^2 is estimated in a similar manner. See the linear regression example in

⁷It applies the original LL method within each block and it is expected to outperform this method.

Section 3.5.1 for further details.

Table 3.1 shows the results for a block size of $N_h = 2$ and a CMP threshold of 0.9. For the intercept and the slope, the magnitude of the relative bias is largest with the naive method as expected. Also, for both parameters, the MSE is smallest with the Complete Data method as expected. For the intercept, the LL estimator has a smaller MSE than the PW estimators. For the slope, the PW estimators have a smaller MSE than the LL estimator. For both parameters, the MSEs of the two PW methods are quite close.

To assess the impact of the block size, it is increased four-fold to $N_h = 8$, while the other parameters are unchanged, including the CMP threshold that is held at 0.9.

Table 3.2 shows the corresponding results.

Table 3.1: Performance under a linear model using linked data from two registers with a block size of $N_h = 2$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	-1.756	0.002972	0.003019
	PW1	-1.675	0.002828	0.00287
	PW2	-1.588	0.002882	0.002916
	LL	-0.611	0.001831	0.001822
	Complete	-0.594	0.00181	0.001801
β_1	Naive	-3.167	0.003223	0.004194
	PW1	-0.004	0.00311	0.003079
	PW2	0.227	0.003097	0.003071
	LL	-0.286	0.005027	0.004985
	Complete	0.313	0.002114	0.002103

Table 3.2: Performance under a linear model using linked data from two registers with a block size of $N_h = 8$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	-0.563	0.000974	0.000973
	PW1	-0.603	0.000998	0.000997
	PW2	-0.605	0.001	0.000999
	LL	-1.032	0.000493	0.000514
	Complete	-1.01	0.000483	0.000504
β_1	Naive	-5.268	0.001498	0.004259
	PW1	0.453	0.001468	0.001474
	PW2	0.446	0.001541	0.001546
	LL	-1.357	0.003825	0.003971
	Complete	-0.065	0.000528	0.000523

Table 3.3: Performance under a linear model using linked data from two registers with a block size of $N_h = 2$ and a CMP threshold of 0.0.

Parameter	Method	Bias (%)	Variance	MSE
β_0	PW1	-0.304	0.001792	0.001777
	PW2	0.007	0.001779	0.001761
β_1	PW1	0.439	0.002221	0.002219
	PW2	0.788	0.003078	0.003109

3.7.3 Logistic model

The regression coefficients are $[\beta_0 \ \beta_1] = [0.5 \ 1]$. For this model, four estimators are considered including the naive estimator, the complete data estimator, and two WLS pairwise estimators according to the methodology of Section 3.5.1. The naive estimator is the solution of the EE

$$\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} l_{ij} \mathbf{x}_i \left(z_j - \frac{e^{\mathbf{x}_i^\top \boldsymbol{\beta}}}{1 + e^{\mathbf{x}_i^\top \boldsymbol{\beta}}} \right) = \mathbf{0}, \quad (3.67)$$

where $\mathbf{x}_i = [1 \ x_i]^\top$ and $l_{ij} = I(q_{ij} \geq q_0)$, where $q_{ij} = P(m_{ij} = 1 | \gamma_{ij})$ and $q_0 = 0.9$.

The complete data estimator is the solution of the unbiased EE

$$\sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} m_{ij} \mathbf{x}_i \left(z_j - \frac{e^{\mathbf{x}_i^\top \boldsymbol{\beta}}}{1 + e^{\mathbf{x}_i^\top \boldsymbol{\beta}}} \right) = \mathbf{0}. \quad (3.68)$$

The first pairwise estimator (later called PW1) is based on the conditional distribution of z_j given γ_{ij} and all the covariates observed in the corresponding block. Like in the linear model, it is the WLS estimator where $v_{ij} = \sigma_{ij}^2$ and $\tau_{ij} = I(q_{ij} \geq q_0)$. The variance σ_{ij}^2 is estimated by using a preliminary estimate of $\boldsymbol{\beta}$ that is based on $v_{ij} = 1$, with the same choice for τ_{ij} . The second pairwise estimator (later called PW2) is based on the conditional distribution of z_j given γ_{ij} and $\mathbf{x}_i$, and the variance σ_{ij} is estimated as for the first pairwise estimator. See the logistic regression example in Section 3.5.1 for further details.

Table 3.4: Performance under a logit model using linked data from two registers with a block size of $N_h = 2$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	-3.961	0.026999	0.027121
	PW1	-3.416	0.028143	0.028153
	PW2	-3.454	0.027941	0.02796
	Complete	-5.62	0.016858	0.017479
β_1	Naive	-5.453	0.085054	0.087177
	PW1	-1.961	0.096278	0.0957
	PW2	-1.884	0.095213	0.094616
	Complete	-1.212	0.061524	0.061056

Table 3.5: Performance under a logit model using linked data from two registers with a block size of $N_h = 8$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	-0.205	0.008951	0.008863
	PW1	0.661	0.00913	0.00905
	PW2	0.696	0.009152	0.009072
	Complete	1.941	0.004053	0.004106
β_1	Naive	-4.676	0.025835	0.027762
	PW1	1.561	0.029943	0.029887
	PW2	1.66	0.029883	0.02986
	Complete	0.618	0.014423	0.014317

Table 3.6: Performance under a logit model using linked data from two registers with a block size of $N_h = 2$ and a CMP threshold of 0.0.

Parameter	Method	Bias (%)	Variance	MSE
β_0	PW1	-1.672	0.023105	0.022944
	PW2	-1.561	0.023638	0.023463
β_1	PW1	-1.611	0.065602	0.065205
	PW2	-0.316	0.068689	0.068012

3.7.4 Survival model

For this model, the responses are survival times that are distributed according to a proportional hazard model, with a constant baseline hazard and censoring. Like in the other models, the coefficients are set to $\beta^\top = [0.5 \ 1]$. The length of the follow-up is $T = 2.0$ with the right-censoring of all survival times exceeding this duration of follow-up. For each individual, register B records the censored survival times as well as an indicator c_j of censoring along with the censored survival time z_j . The naive estimator is as follows.

$$\hat{\beta} = \arg \max_{\beta} \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} l_{ij} \log \left((1 - c_j) e^{\mathbf{x}_i^\top \beta} \exp \left(-e^{\mathbf{x}_i^\top \beta} z_j \right) + c_j \exp \left(-e^{\mathbf{x}_i^\top \beta} T \right) \right), \quad (3.69)$$

where $\mathbf{x}_i = [1 \ x_i]^\top$, $l_{ij} = I(q_{ij} \geq q_0)$, where $q_{ij} = P(m_{ij} = 1 | \gamma_{ij})$ and $q_0 = 0.9$. The complete data estimator is the solution of the unbiased EE

$$\hat{\beta} = \arg \max_{\beta} \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} m_{ij} \log \left((1 - c_j) e^{\mathbf{x}_i^\top \beta} \exp \left(-e^{\mathbf{x}_i^\top \beta} z_j \right) + c_j \exp \left(-e^{\mathbf{x}_i^\top \beta} T \right) \right). \quad (3.70)$$

The pairwise estimators are based on the maximum likelihood as described in Section 3.5.2. The first PW estimator (PW1) is based on the conditional distribution of

z_j given γ_{ij} and all the covariates in the corresponding block. The second PW estimator (PW2) is based on the conditional distribution of z_j given γ_{ij} and $\mathbf{x}_i = [1 \ x_i]^\top$. See the survival model example in Section 3.5.2 for further details. When $N_h = 8$, PW1 performs better than PW2 because in PW1 the conditioning event is based all the block covariates instead of covariates from a single record, as in PW2.

Table 3.7: Performance under an exponential PHM using linked data from two registers with a block size of with $N_h = 2$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	4.679	0.012439	0.012862
	PW1	1.319	0.012284	0.012204
	PW2	1.472	0.012182	0.012115
	Complete	-0.111	0.010338	0.010235
β_1	Naive	-6.471	0.008717	0.012817
	PW1	-0.374	0.007031	0.006975
	PW2	-0.496	0.007242	0.007195
	Complete	-0.166	0.004796	0.004751

3.7.5 Conclusions

Overall, the results for the different models show the following. The magnitude of the relative bias is typically smaller with PW1 and PW2 than with the naive estimator. In fact, it is often much smaller. The results also show that PW1 and PW2 have a similar MSE performance, with a slight advantage for PW1 over PW2. Finally, their MSE performance is improved by a larger block size and by a lower CMP threshold,

Table 3.8: Performance under an exponential PHM using linked data from two registers with a block size of with $N_h = 8$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	2.153	0.010718	0.010727
	PW1	-0.981	0.010918	0.010833
	PW2	-0.996	0.011031	0.010946
	Complete	-0.202	0.007736	0.007659
β_1	Naive	-5.095	0.007634	0.010153
	PW1	1.199	0.006101	0.006184
	PW2	1.205	0.006131	0.006215
	Complete	0.612	0.003806	0.003806

Table 3.9: Performance under an exponential PHM using linked data from two registers with a block size of with $N_h = 2$ and a CMP threshold of 0.0.

Parameter	Method	Bias (%)	Variance	MSE
β_0	PW1	0.101	0.00922	0.009128
	PW2	-0.036	0.009823	0.009725
β_1	PW1	0.191	0.004144	0.004106
	PW2	0.25	0.004696	0.004655

all other things being equal.

Chapter 4

Pairwise EEs when linking samples

4.1 Overview

In this chapter, we consider two data sources that include a sample and a register, or two overlapping samples from the same finite population. The two samples may be registers that have some undercoverage. The following sections are organized as follows. Section 4.2 gives the notation. Section 4.3 lists the assumptions. Section 4.4 derives various conditional means of the observed responses. Section 4.5 describes the proposed estimation procedures. Section 4.6 discusses the large sample properties of the proposed estimators. Section 4.7 describes a simulation study.

4.2 Notation

Without losing any generality, identify file A with a sample drawn from a first register A^* (identified with $A^* = \{1 \dots N\}$) with complete coverage of the finite population. In this register, file A records are indexed over a subset $A \subset \{1, \dots, N\}$. In a similar manner, identify file B with a sample drawn from a second register B^* (identified with $B^* = \{1 \dots N\}$), which also has a complete coverage of the same population. File

B records are indexed over a subset $B \subset \{1, \dots, N\}$, where $|B|$ is possibly different from $|A|$. We are interested in situations where the two files overlap significantly, i.e., where the ratio $|A \cap B| / \min(|A|, |B|)$ is sufficiently large¹. In the Cartesian product $A^* \times B^*$, consider the pair (i, j) and define m_{ij} and γ_{ij} , the pair match status and comparison vector, respectively. Let $\mathbf{M} = [m_{ij}]_{1 \leq i, j \leq N}$ denote the match matrix in $A^* \times B^*$. In B^* , let $\mathbf{z}_j$ denote the observed responses from record j in B^* , and define the vector $\mathbf{z} = [\mathbf{z}_1^\top \dots \mathbf{z}_N^\top]^\top$. As before, let $\mathbf{X} = [\mathbf{x}_1^\top \dots \mathbf{x}_N^\top]^\top$ denote the matrix of all the covariates in register A^* . Finally let $\mathbf{y} = [\mathbf{y}_1^\top \dots \mathbf{y}_N^\top]^\top$ denote the actual responses, where $\mathbf{y}_i$ is the actual response for record i in A^* . As before, the finite population comprises of H IID blocks that each contain a variable but bounded number of IID individuals. Block h has size N_h , where $N_h \leq C$ for some constant C that does not depend on H , with $N = N_1 + \dots + N_H$. The block also corresponds to records indexed in the subsets A_h^* and B_h^* in the files A^* and B^* respectively, where $|A_h^*| = |B_h^*| = N_h$. Let A_h and B_h denote the corresponding subsets in files A and B respectively, and let $\mathbf{M}_h$ denote the match matrix in $A_h^* \times B_h^*$; the Cartesian product within the block.

4.3 Assumptions

The following assumptions are made that extend those of Section 3.3.

- A.1 The match matrix $\mathbf{M}_h$ is a uniform random permutation independent of $[\mathbf{x}_i]_{i \in A_h^*}$.
- A.2 For $i \in A_h^*$, let $j(i)$ denote the index of the corresponding record in B_h^* . The variables $\left([\mathbf{y}_i]_{i \in A_h^*}, [I(j(i) \in B_h)]_{i \in A_h^*}, [I(i \in A_h)]_{i \in A_h^*}, \mathbf{M}_h, \text{ and } \{\gamma_{ij}\}_{(i,j) \in A_h^* \times B_h^*}\right)$ are conditionally mutually independent given the block size N_h and the covariates $[\mathbf{x}_i]_{i \in A_h^*}$.

¹When the overlap is small, statistical matching may be a better solution.

A.3 The conditional match probability $P\left(m_{i'j} = 1 \mid [\mathbf{x}_i]_{i \in A_h^*}, N_h, \gamma_{ij}\right)$ is the same for all $i' \in A_h^* - \{i\}$.

A.4 The sequence $[(\mathbf{x}_i, \mathbf{y}_i, I(i \in A_h), I(j(i) \in B_h))]_{i \in A_h^*}$ is IID given the block size N_h , with a common distribution that does not depend on N_h .

A.5 In block h , for each pair (i, j) , the variables (m_{ij}, γ_{ij}) , $[\mathbf{x}_{i''}]_{i'' \in A_h^* - \{i\}}$, and $I(i \in A_h)$ are conditionally independent given the block size N_h , and the pair covariates $\mathbf{x}_i$.

Assumption A.5 implies the following identity:

$$\begin{aligned}
 P\left(m_{ij} = 1 \mid [\mathbf{x}_{i'}]_{i' \in A_h^*}, N_h, \gamma_{ij}\right) &= \frac{P\left(m_{ij} = 1 \text{ and } \gamma_{ij} \mid N_h, \mathbf{x}_i, [\mathbf{x}_{i'}]_{i' \in A_h^* - \{i\}}\right)}{P\left(\gamma_{ij} \mid N_h, \mathbf{x}_i, [\mathbf{x}_{i'}]_{i' \in A_h^* - \{i\}}\right)} \\
 &= \frac{P\left(m_{ij} = 1 \text{ and } \gamma_{ij} \mid N_h, \mathbf{x}_i\right)}{P\left(\gamma_{ij} \mid N_h, \mathbf{x}_i\right)} \\
 &= P\left(m_{ij} = 1 \mid \mathbf{x}_i, N_h, \gamma_{ij}\right) \\
 &= \left(1 + \left(\frac{1}{N_h} - 1\right) \frac{P(\gamma_{ij} \mid \mathbf{x}_i, m_{ij} = 0)}{P(\gamma_{ij} \mid \mathbf{x}_i, m_{ij} = 1)}\right)^{-1}.
 \end{aligned} \tag{4.1}$$

4.4 Conditional response distribution

In this section, we extend the results of Section 3.4 by accounting for different missing data mechanisms. It gives the conditional mean of $\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)$, given that the pair (i, j) is selected in $A_h \times B_h$, the comparison vector γ_{ij} and the observed covariates in A_h (including $\mathbf{x}_i$).

4.4.1 Information from a block

Corollary 5 is the main result of this section and Theorem 5 is a key stepping stone, where only the covariates are missing.

Theorem 5 *Consider a fixed nonempty subset s_h of A_h^* , and $(i, j) \in s_h \times B_h^*$. Suppose that assumptions A.1-A.5 (from Section 4.3) apply. For some individual i'' in block h , let $\pi(\mathbf{x}_{i''})$ denote the probability of selection of individual i'' in file A , i.e. $P(i'' \in A_h | \mathbf{x}_{i''})$, and define the event $O_{ij} = \{N_h\} \cap \{[\mathbf{x}_{i''}]_{i'' \in s_h}\} \cap \{\gamma_{ij}\} \cap \{A_h = s_h\}$. Then*

$$\begin{aligned} E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}] &= q_{ij} E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] + \\ &\quad (1 - q_{ij}) \left(\frac{|s_h| - 1}{N_h - 1} \frac{\sum_{i' \in s_h - \{i\}} E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}]}{|s_h| - 1} + \right. \\ &\quad \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | \mathbf{x}_i]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right), \end{aligned} \quad (4.2)$$

where q_{ij} is given by Eq. (4.1).

Proof: Consider $(i, j) \in A_h^* \times B_h^*$. As before, we have

$$\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) = \sum_{i' \in A_h^*} m_{i'j} \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i).$$

Hence

$$\begin{aligned} &E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij}, A_h = s_h] \\ &= \sum_{i' \in A_h^*} E[m_{i'j} \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij}, A_h = s_h] \end{aligned}$$

Using the conditional independence of $\left(\mathbf{M}_h, [\gamma_{i''j''}]_{(i'',j'') \in A_h^* \times B_h^*}\right)$, $[\mathbf{y}_{i''}]_{i'' \in B_h^*}$ and A_h given N_h and $[\mathbf{x}_{i''}]_{i'' \in A_h^*}$, we have

$$\begin{aligned}
& E \left[m_{i'j} \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij}, A_h = s_h \right] \\
&= E \left[m_{i'j} \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij} \right] \\
&= E \left[m_{i'j} \mid [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij} \right] E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*} \right] \\
&= P \left(m_{i'j} = 1 \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij} \right) E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right]
\end{aligned}$$

Therefore,

$$\begin{aligned}
& E \left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij}, A_h = s_h \right] \\
&= \sum_{i' \in A_h^*} P \left(m_{i'j} = 1 \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij} \right) E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \\
&= P \left(m_{ij} = 1 \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij} \right) E \left[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mid \mathbf{x}_i \right] + \\
&\quad \sum_{i' \in A_h^* - \{i\}} P \left(m_{i'j} = 1 \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij} \right) E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \\
&= q_{ij} E \left[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mid \mathbf{x}_i \right] + \frac{q_{ij}}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right],
\end{aligned}$$

where the last equation follows from Eq. (3.3) and assumption A.5.

Then

$$\begin{aligned}
& E \left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right] \\
&= E \left[q_{ij} E \left[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mid \mathbf{x}_i \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right] + \\
& \frac{1}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} E \left[(1 - q_{ij}) E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right] \\
&= q_{ij} E \left[E \left[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mid \mathbf{x}_i \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right] + \\
& \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right] \\
&= q_{ij} E \left[E \left[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mid \mathbf{x}_i \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right] + \\
& \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in s_h - \{i\}} E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right] + \\
& \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h^* - s_h} E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right], \quad (4.3)
\end{aligned}$$

because q_{ij} is a constant function given $\mathbf{x}_i$ and γ_{ij} , and $E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}]$ is also a constant function given $[\mathbf{x}_{i''}]_{i'' \in s_h}$ when $\{i, i'\} \subset s_h$. Now, consider $i' \in A_h^* - s_h$ and write

$$\begin{aligned}
& E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h \right] \\
&= \frac{E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] I(\gamma_{ij}, A_h = s_h) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h} \right]}{E \left[I(\gamma_{ij}, A_h = s_h) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h} \right]}. \quad (4.4)
\end{aligned}$$

Again, using the conditional independence of $\left(\mathbf{M}_h, [\gamma_{i''j''}]_{(i'',j'') \in A_h^* \times B_h^*}\right)$, $[\mathbf{y}_{i''}]_{i'' \in B_h^*}$ and A_h given N_h and $[\mathbf{x}_{i''}]_{i'' \in A_h^*}$, we have

$$\begin{aligned}
& E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] I(\gamma_{ij}, A_h = s_h) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*} \right] \\
&= E \left[I(\gamma_{ij}) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*} \right] E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*} \right] \times \\
&\quad E \left[I(A_h = s_h) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*} \right] \\
&= E \left[I(\gamma_{ij}) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*} \right] E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \prod_{i'' \in A_h^*} \pi(\mathbf{x}_{i''})^{I(i'' \in s_h)} (1 - \pi(\mathbf{x}_{i''}))^{I(i'' \notin s_h)} \\
&= P(\gamma_{ij} \mid N_h, \mathbf{x}_i) E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \prod_{i'' \in A_h^*} \pi(\mathbf{x}_{i''})^{I(i'' \in s_h)} (1 - \pi(\mathbf{x}_{i''}))^{I(i'' \notin s_h)}.
\end{aligned}$$

Then

$$\begin{aligned}
& E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] I(\gamma_{ij}, A_h = s_h) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h} \right] \\
&= E \left[P(\gamma_{ij} \mid N_h, \mathbf{x}_i) E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \times \right. \\
&\quad \left. \prod_{i'' \in A_h^*} \pi(\mathbf{x}_{i''})^{I(i'' \in s_h)} (1 - \pi(\mathbf{x}_{i''}))^{I(i'' \notin s_h)} \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h} \right] \\
&= E \left[\prod_{i'' \in A_h^* - (s_h \cup \{i'\})} (1 - \pi(\mathbf{x}_{i''})) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h} \right] \times \\
&\quad E \left[P(\gamma_{ij} \mid N_h, \mathbf{x}_i) E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \times \right. \\
&\quad \left. \prod_{i'' \in s_h \cup \{i'\}} \pi(\mathbf{x}_{i''})^{I(i'' \in s_h)} (1 - \pi(\mathbf{x}_{i''}))^{I(i'' \notin s_h)} \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h} \right] \\
&= E \left[(1 - \pi(\mathbf{x}_{i''})) \right]^{N_h - |s_h| - 1} P(\gamma_{ij} \mid N_h, \mathbf{x}_i) \left(\prod_{i'' \in s_h} \pi(\mathbf{x}_{i''}) \right) \times \\
&\quad E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] (1 - \pi(\mathbf{x}_{i'})) \mid \mathbf{x}_i \right]. \tag{4.5}
\end{aligned}$$

Using similar arguments, we have

$$\begin{aligned}
& E \left[I(\gamma_{ij}, A_h = s_h) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s_h} \right] \\
&= E \left[(1 - \pi(\mathbf{x}_{i''})) \right]^{N_h - |s_h| - 1} P(\gamma_{ij} \mid N_h, \mathbf{x}_i) \left(\prod_{i'' \in s_h} \pi(\mathbf{x}_{i''}) \right) E \left[(1 - \pi(\mathbf{x}_{i'})) \mid N_h, \mathbf{x}_i \right] \\
&= E \left[(1 - \pi(\mathbf{x}_{i''})) \right]^{N_h - |s_h| - 1} P(\gamma_{ij} \mid N_h, \mathbf{x}_i) \left(\prod_{i'' \in s_h} \pi(\mathbf{x}_{i''}) \right) E \left[(1 - \pi(\mathbf{x}_{i'})) \right] \tag{4.6}
\end{aligned}$$

Then use Eq. (4.5) and Eq. (4.6) to rewrite Eq. (4.4) as follows

$$\begin{aligned}
 & E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] \mid N_h, [\mathbf{x}_{i''}]_{i'' \in s}, \gamma_{ij}, A_h = s_h \right] \\
 &= \frac{E \left[E \left[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'} \right] (1 - \pi(\mathbf{x}_{i'})) \mid \mathbf{x}_i \right]}{E \left[(1 - \pi(\mathbf{x}_{i'})) \right]},
 \end{aligned}$$

where $i' \in A_h^* - s_h$. The above equation and Eq. (4.3) lead to the conclusion.

Q.E.D.

The following corollary is a straightforward extension of the previous theorem for missing responses.

Corollary 5 *Consider a fixed nonempty subset s_h of A_h^* , and $(i, j) \in s_h \times B_h^*$. Suppose that assumptions A.1-A.5 (from Section 4.3) apply. For some individual i'' in block h , let $\pi(\mathbf{x}_{i''})$ denote the probability of selection of individual i'' in file A , i.e. $P(i'' \in A_h \mid \mathbf{x}_{i''})$, and define the event*

$$O_{ij} = \{N_h\} \cap \{[\mathbf{x}_{i''}]_{i'' \in s_h}\} \cap \{\gamma_{ij}\} \cap \{A_h = s_h\} \cap \{j \in B_h\}.$$

Then

$$\begin{aligned}
& E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}] \\
&= \frac{\left[q_{ij} E[I(j(i) \in B_h) \mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] + \right. \\
&\quad \left. (1 - q_{ij}) \left(\frac{\sum_{i' \in s - \{i\}} E[I(j(i') \in B_h) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}]}{N_h - 1} + \right. \right. \\
&\quad \left. \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) E[I(j(i') \in B_h) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | \mathbf{x}_i]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}{\left[q_{ij} E[I(j(i) \in B_h) | \mathbf{x}_i] + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} E[I(j(i') \in B_h) | \mathbf{x}_{i'}]}{N_h - 1} + \right. \right. \\
&\quad \left. \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) E[I(j(i') \in B_h) | \mathbf{x}_{i'}]]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]} \tag{4.7}
\end{aligned}$$

where q_{ij} is given by Eq. (4.1).

Proof: First note that

$$\begin{aligned}
& E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h, j \in B_h] \\
&= \frac{E[I(j \in B_h) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h]}{E[I(j \in B_h) | N_h, [\mathbf{x}_{i''}]_{i'' \in s_h}, \gamma_{ij}, A_h = s_h]}.
\end{aligned}$$

Conclude by applying Theorem 5 to the numerator (with $\mathbf{g}(\mathbf{y}_j, \mathbf{x}_i)$ replaced by $I(j(i) \in B_h) \mathbf{g}(\mathbf{y}_j, \mathbf{x}_i)$) and to the denominator (with $\mathbf{g}(\mathbf{y}_j, \mathbf{x}_i)$ replaced by $I(j(i) \in B_h)$).

Q.E.D.

The next corollary is a straightforward application of Corollary 5. It may be proved as Corollary 3.

Corollary 6 Consider a fixed nonempty subset s_h of A_h^* , and $(i, j) \in s_h \times B_h^*$. Suppose that assumptions A.1-A.5 (from Section 4.3) apply. For some individual i'' in block h , let $\pi(\mathbf{x}_{i''})$ denote the probability of selection of individual i'' in file A , i.e. $P(i'' \in A_h | \mathbf{x}_{i''})$, and define the event

$$O_{ij} = \{N_h\} \cap \{\mathbf{x}_{i''}\}_{i'' \in s_h\} \cap \{\gamma_{ij}\} \cap \{A_h = s_h\} \cap \{j \in B_h\}.$$

Let Σ_{ij} denote the conditional variance-covariance of $\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)$. Then

$$\Sigma_{ij} = E \left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)^\top \middle| O_{ij} \right] - E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}] E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}]^\top, \quad (4.8)$$

where $E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}]$ is given by Eq. (4.7), q_{ij} is given by Eq. (4.1) and

$$\begin{aligned} & E \left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)^\top \middle| O_{ij} \right] \\ &= \frac{\left[q_{ij} E \left[I(j(i) \in B_h) \mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mathbf{g}(\mathbf{y}_i, \mathbf{x}_i)^\top \middle| \mathbf{x}_i \right] + \right. \\ & \quad \left. (1 - q_{ij}) \left(\frac{\sum_{i' \in s - \{i\}} E \left[I(j(i') \in B_h) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i)^\top \middle| \mathbf{x}_i, \mathbf{x}_{i'} \right]}{N_h - 1} + \right. \right. \\ & \quad \left. \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E \left[(1 - \pi(\mathbf{x}_{i'})) E \left[I(j(i') \in B_h) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i)^\top \middle| \mathbf{x}_i, \mathbf{x}_{i'} \right] \middle| \mathbf{x}_i \right]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}{\left[q_{ij} E[I(j(i) \in B_h) | \mathbf{x}_i] + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} E[I(j(i') \in B_h) | \mathbf{x}_{i'}]}{N_h - 1} + \right. \right. \\ & \quad \left. \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) E[I(j(i') \in B_h) | \mathbf{x}_{i'}]]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]} \end{aligned} \quad (4.9)$$

The next corollary is another straightforward application of Corollary 5.

Corollary 7 Consider a fixed nonempty subset s_h of A_h^* , and $(i, j) \in s_h \times A_h^*$. Let $\pi(\mathbf{x}_{i''}) = P(i'' \in A_h | \mathbf{x}_{i''})$ and let $\pi(\mathbf{x}_{i'}) = P(i' \in A_h | \mathbf{x}_{i'})$. Let $f_{y|x \cap B}(\cdot | \cdot)$ denote the conditional PDF or PMF of $\mathbf{y}_{i''}$ given $\mathbf{x}_{i''}$ and $j(i'') \in B_h$. Also, let O_{ij} denote the event $\{N_h\} \cap \{\mathbf{x}_{i''}\}_{i'' \in s_h} \cap \{A_h = s_h\} \cap \{j \in B_h\}$, and let $f_{ij}(\cdot | \cdot)$ denote the conditional PDF or PMF of $\mathbf{z}_j$ given O_{ij} . Then

$$f_{ij}(\boldsymbol{\zeta} | O_{ij}) = \frac{\left[q_{ij} P(j(i) \in B_h | \mathbf{x}_i) f_{y|x \cap B}(\boldsymbol{\zeta} | \mathbf{x}_i) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} P(j(i') \in B_h | \mathbf{x}_{i'}) f_{y|x \cap B}(\boldsymbol{\zeta} | \mathbf{x}_{i'})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) P(j(i') \in B_h | \mathbf{x}_{i'}) f_{y|x \cap B}(\boldsymbol{\zeta} | \mathbf{x}_{i'}) | \mathbf{x}_i]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}{\left[q_{ij} P(j(i) \in B_h | \mathbf{x}_i) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} P(j(i') \in B_h | \mathbf{x}_{i'})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) P(j(i') \in B_h | \mathbf{x}_{i'})]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]} \quad (4.10)$$

Proof: Simply apply Corollary 5 with $g(\mathbf{y}_i, \mathbf{x}_i) = I(\mathbf{y}_i \leq \boldsymbol{\zeta})$ to obtain the conditional CDF of $\mathbf{z}_j$. Then obtain the density (PDF for a continuous response and PMF for a categorical response) as the Radon-Nikodym derivative [46, Section 32, pp. 423] of the CDF.

Q.E.D.

We next apply the above results to some examples. They serve to illustrate some of the differences between regression with linked data and classical regression with missing responses or covariates.

Responses that are Missing at Random (MAR): Suppose that file A is complete (i.e., $A_h = A_h^*$) but that the responses are MAR in file B, i.e., $\mathbf{y}_i$ and $I(j(i) \in B_h)$ are conditionally independent given $\mathbf{x}_i$. Let $\nu(\mathbf{x}_i) = P(j(i) \in B_h | \mathbf{x}_i)$ denote the probability

that the response is recorded in file B. Then

$$E \left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i'}]_{i' \in S}, \gamma_{ij}, A_h = s_h, j \in B_h \right] = \frac{q_{ij} \nu(\mathbf{x}_i) E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mid \mathbf{x}_i] + \frac{1 - q_{ij}}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} \nu(\mathbf{x}_{i'}) E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}]}{q_{ij} \nu(\mathbf{x}_i) + \frac{1 - q_{ij}}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} \nu(\mathbf{x}_{i'})}. \quad (4.11)$$

The above result has a simple interpretation. The observed covariates are weighted according to their response propensity that gives a measure of the likelihood that they have produced the given response. This equation shows that the response propensity $[\nu(\mathbf{x}_{i'})]_{i'}$ cannot be ignored, unlike what happens in common regression problems with MAR responses. This comment also applies to the situation where the response file is based on a survey with a complex design, where $\nu(\mathbf{x}_i)$ is related to the design weights. In other words, the analysis of the linked data must incorporate the design weights even if the sample design is noninformative, i.e., the inclusion in the sample ($I(j(i) \in B_h)$) is unrelated to the response $\mathbf{y}_i$ given the covariates $\mathbf{x}_i$.

Survival model: Suppose that file B records events that occur by some time T , including the time of each such occurrence. In this case, the response y_i is simply the occurrence time t_i . Then the event $\{j(i) \in B_h\}$ is equivalent to the event $\{t_i \leq T\}$. The event time is not recorded in file B if $t_i > T$. Thus the responses are not missing at random (NMAR). In this case, Eq. (4.10) takes the following simple form, where $F(\cdot \mid \mathbf{x}_i)$ and $f(\cdot \mid \mathbf{x}_i)$ denote the conditional CDF and PDF of the occurrence time

respectively and $t \leq T$:

$$f_{ij}(t|O_{ij}) = \frac{\left[q_{ij}f(t|\mathbf{x}_i) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} f(t|\mathbf{x}_{i'})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'}))f(t|\mathbf{x}_{i'})|\mathbf{x}_i]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}{\left[q_{ij}F(T|\mathbf{x}_i) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} F(T|\mathbf{x}_{i'})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'}))F(T|\mathbf{x}_{i'})]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}. \quad (4.12)$$

4.4.2 Information from a single pair

The results of Section 4.4.1 require the block size N_h of every block. This is a problem when these variables are not directly observed such as when linking two samples. A convenient alternative only considers the information from a selected pair, i.e. a pair selected in $A_h \times B_h$. It is based on Corollaries 8, 9 and 10 hereafter that extend the related results in Section 3.4.2. These corollaries are all straightforward applications of Theorem 6; the first result in this section.

Theorem 6 *Suppose that assumptions A.1-A.5 (from Section 4.3) apply. Consider (i, j) in block h and define the event $O_{ij} = \{N_h\} \cap \{\mathbf{x}_i\} \cap \{\gamma_{ij}\} \cap \{i \in A_h\}$. Then*

$$\begin{aligned} E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)|O_{ij}] &= E[q_{ij}|O_{ij}] E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i)|\mathbf{x}_i] + \\ &\quad (1 - E[q_{ij}|O_{ij}]) E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i)|\mathbf{x}_i], \end{aligned} \quad (4.13)$$

where q_{ij} is given by Eq. (4.1).

Proof: Consider $(i, j) \in A_h^* \times B_h^*$. As before, we have

$$\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) = \sum_{i' \in A_h^*} m_{i'j} \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i).$$

Hence

$$\begin{aligned} & E \left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij}, i \in A_h \right] \\ &= \sum_{i' \in A_h^*} E \left[m_{i'j} \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij}, i \in A_h \right]. \end{aligned}$$

As in the proof of Theorem 5, use the conditional independence (assumption A.2 in Section 4.3) of $(\mathbf{M}_h, [\gamma_{i''j''}]_{(i'', j'') \in A_h^* \times B_h^*})$, $[\mathbf{y}_{i''}]_{i'' \in B_h^*}$ and A_h given N_h and $[\mathbf{x}_{i''}]_{i'' \in A_h^*}$, to get the following identity:

$$\begin{aligned} & E \left[m_{i'j} \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij}, i \in A_h \right] \\ &= E \left[m_{i'j} \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij} \right]. \end{aligned}$$

Next, proceed as in the proof of Theorem 4.4.1-1 to obtain Eq. (4.3) and the following identity:

$$\begin{aligned} & E \left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mid N_h, [\mathbf{x}_{i''}]_{i'' \in A_h^*}, \gamma_{ij}, i \in A_h \right] \\ &= q_{ij} E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mid \mathbf{x}_i] + \frac{q_{ij}}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mid \mathbf{x}_i, \mathbf{x}_{i'}]. \end{aligned} \quad (4.14)$$

Consequently,

$$\begin{aligned}
& E[g(\mathbf{z}_j, \mathbf{x}_i) | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h] \\
&= E[q_{ij} E[g(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h] + \\
& \quad \frac{1}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} E[(1 - q_{ij}) E[g(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h] \\
&= q_{ij} E[E[g(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h] + \\
& \quad \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} E[E[g(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h] \\
&= q_{ij} E[g(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] + \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} E[E[g(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h].
\end{aligned} \tag{4.15}$$

Using the conditional independence of (m_{ij}, γ_{ij}) , $[\mathbf{x}_{i'']}_{i'' \in A_h^* - \{i\}}$ and $I(i \in A_h)$ given N_h and $\mathbf{x}_i$ (assumption A.5 in Section 4.3), we have

$$\begin{aligned}
E[E[g(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h] &= E[E[g(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i, \mathbf{x}_{i'}] | N_h, \mathbf{x}_i] \\
&= E[g(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i],
\end{aligned}$$

where the last equation is a consequence of assumption A.4 in Section 4.3. Hence

$$\begin{aligned}
E[g(\mathbf{z}_j, \mathbf{x}_i) | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h] &= q_{ij} E[g(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] + \\
& \quad \frac{(1 - q_{ij})}{N_h - 1} \sum_{i' \in A_h^* - \{i\}} E[g(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i] \\
&= q_{ij} E[g(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] + \\
& \quad (1 - q_{ij}) E[g(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i],
\end{aligned}$$

because $E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i]$ is constant over $A_h^* - \{i\}$. Thus

$$\begin{aligned}
 E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | \mathbf{x}_i, \gamma_{ij}, i \in A_h] &= E[q_{ij} E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] | \mathbf{x}_i, \gamma_{ij}, i \in A_h] + \\
 &\quad E[(1 - q_{ij}) E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i] | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \\
 &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] + \\
 &\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i],
 \end{aligned}$$

where the last equation is based on the fact that both $E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i]$ and $E[\mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i]$ are functions of $\mathbf{x}_i$ (only).

Q.E.D.

Corollary 8 Suppose that assumptions A.1-A.5 (from Section 4.3) apply. Consider (i, j) in block h and define the event $O_{ij} = \{N_h\} \cap \{\mathbf{x}_i\} \cap \{\gamma_{ij}\} \cap \{i \in A_h\} \cap \{j \in B_h\}$. Then

$$\begin{aligned}
 &E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}] \\
 &= \frac{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] E[I(j(i) \in B_h) \mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) | \mathbf{x}_i] + \right. \\
 &\quad \left. (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[I(j(i') \in B_h) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) | \mathbf{x}_i] \right)}{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] P(j(i) \in B_h | \mathbf{x}_i) + \right. \\
 &\quad \left. (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[P(j(i') \in B_h)] \right)}.
 \end{aligned} \tag{4.16}$$

Proof: Proceeding as in Corollary 5, first note that

$$\begin{aligned} & E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h, j \in B_h] \\ &= \frac{E[I(j \in B_h) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h]}{E[I(j \in B_h) | N_h, \mathbf{x}_i, \gamma_{ij}, i \in A_h]}. \end{aligned}$$

Next apply Theorem 6 to the numerator and denominator separately.

Q.E.D.

Corollary 9 Suppose that assumptions A.1-A.5 (from Section 4.3) apply. For the pair (i, j) in block h define the event $O_{ij} = \{N_h\} \cap \{\mathbf{x}_i\} \cap \{\gamma_{ij}\} \cap \{i \in A_h\} \cap \{j \in B_h\}$ and let Σ_{ij} denote the conditional variance-covariance of $\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)$. Then

$$\Sigma_{ij} = E\left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)^\top \middle| O_{ij}\right] - E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}] E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}]^\top,$$

where $E[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) | O_{ij}]$ is given by Eq. (4.16) and

$$\begin{aligned} & E\left[\mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)^\top \middle| O_{ij}\right] \\ &= \frac{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] E\left[I(j(i) \in B_h) \mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \mathbf{g}(\mathbf{y}_i, \mathbf{x}_i)^\top \middle| \mathbf{x}_i\right] + \right. \\ & \quad \left. (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E\left[I(j(i') \in B_h) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i) \mathbf{g}(\mathbf{y}_{i'}, \mathbf{x}_i)^\top \middle| \mathbf{x}_i\right] \right)}{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] P(j(i) \in B_h | \mathbf{x}_i) + \right. \\ & \quad \left. (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[P(j(i') \in B_h)] \right)}. \end{aligned} \tag{4.17}$$

Corollary 10 Suppose that assumptions A.1-A.5 (from Section 4.3) apply. For the

pair (i, j) in block h define the event $O_{ij} = \{N_h\} \cap \{\mathbf{x}_i\} \cap \{\gamma_{ij}\} \cap \{i \in A_h\} \cap \{j \in B_h\}$ and let $f_{y|x \cap B}(\cdot|\cdot)$ denote the conditional PDF or PMF of $\mathbf{y}_{i'}$ given $\mathbf{x}_{i'}$ and $j(i'') \in B_h$. Also let $f_{ij}(\cdot|\cdot)$ denote the conditional PDF or PMF of $\mathbf{z}_j$ given O_{ij} . Then

$$f_{ij}(\boldsymbol{\zeta}|O_{ij}) = \frac{\left(E[q_{ij}|\mathbf{x}_i, \gamma_{ij}, i \in A_h] P(j(i) \in B_h|\mathbf{x}_i) f_{y|x \cap B}(\boldsymbol{\zeta}|\mathbf{x}_i) + \right. \\ \left. (1 - E[q_{ij}|\mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[P(j(i') \in B_h|\mathbf{x}_{i'}) f_{y|x \cap B}(\boldsymbol{\zeta}|\mathbf{x}_{i'})] \right)}{\left(E[q_{ij}|\mathbf{x}_i, \gamma_{ij}, i \in A_h] P(j(i) \in B_h|\mathbf{x}_i) + \right. \\ \left. (1 - E[q_{ij}|\mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[P(j(i') \in B_h|\mathbf{x}_{i'})] \right)} \quad (4.18)$$

Proof: Apply Corollary 8 with $g(\mathbf{y}_i, \mathbf{x}_i) = I(\mathbf{y}_i \leq \boldsymbol{\zeta})$ to obtain the conditional CDF of $\mathbf{z}_j$. Then obtain the density (PDF for a continuous response and PMF for a categorical response) as the Radon-Nikodym derivative [46, Section 32, pp. 423] of the CDF.

Q.E.D.

4.5 Estimation procedures

In the following sections, we extend the estimators described in Section 3.5.

4.5.1 Weighted Least Squares

As before, consider a general regression model $E[\mathbf{y}_i|\mathbf{x}_i] = \mu(\mathbf{x}_i; \boldsymbol{\theta}_0)$, where $\boldsymbol{\theta}_0$ is unknown but $\mu(\cdot, \cdot)$ is known, and let $\mathbf{g}(\mathbf{z}_j; \mathbf{x}_i, \boldsymbol{\theta}) = \mathbf{z}_j$. We make all the assumptions of Section 4.3, and further assume that the responses $[\mathbf{y}_i]_{i \in A_h^*}$ and the file B inclusion

indicators $[I(j(i) \in B_h)]_{i \in A_h^*}$ are conditionally independent given the block size N_h and the covariates $[\mathbf{x}_i]_{i \in A_h^*}$. The estimation procedure of Section 3.5.1 still applies. In this methodology, the estimator $\hat{\boldsymbol{\theta}}$ is the solution of Eq. (3.19). However some changes are required, especially regarding $\boldsymbol{\Delta}_{ij}$, the choice of the matrix $\mathbf{V}_{ij}$ (in Eq. (3.19)) and the nuisance parameters, which now include parameters related to the *recording propensities* in the two files. We next describe these changes.

In Eq. (3.16), which gives the general expression for $\boldsymbol{\Delta}_{ij}$, the conditional expectation $E[\mathbf{z}_j | O_{ij}]$ is now based on Corollary 5 or Corollary 8. According to Corollary 5, $E[\mathbf{z}_j | O_{ij}]$ is

$$E[\mathbf{z}_j | O_{ij}] = \frac{\left[q_{ij} \nu(\mathbf{x}_i) \mu(\mathbf{x}_i; \boldsymbol{\theta}) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s - \{i\}} \nu(\mathbf{x}_{i'}) \mu(\mathbf{x}_{i'}; \boldsymbol{\theta})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mu(\mathbf{x}_{i'}; \boldsymbol{\theta})]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}{\left[q_{ij} \nu(\mathbf{x}_i) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \nu(\mathbf{x}_{i'})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'})]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}, \quad (4.19)$$

where $\nu(\mathbf{x}_i) = P(j(i) \in B_h | \mathbf{x}_i)$ and $\pi(\mathbf{x}_i) = P(i \in A_h | \mathbf{x}_i)$.

According to Corollary 8, $E[\mathbf{z}_j | O_{ij}]$ is

$$E[\mathbf{z}_j | O_{ij}] = \frac{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) \mu(\mathbf{x}_i, \boldsymbol{\theta}) + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'}) \mu(\mathbf{x}_{i'}, \boldsymbol{\theta})] \right)}{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'})] \right)}. \quad (4.20)$$

The choice $\mathbf{V}_{ij} = \Sigma_{ij} = E[\Delta_{ij}\Delta_{ij}^\top | O_{ij}]$ is still quasi-optimal², but Σ_{ij} is now based on Corollary 6 or Corollary 9. This conditional variance-covariance may require the estimation of variance components.

The nuisance parameters now include ψ , the variance components, and additional parameters that are related to the marginal distribution of the covariates or the recording propensities for the two files. When the estimation procedure is based on Corollary 5, the additional parameters are required to estimate the expectations $E[(1 - \pi(\mathbf{x}_{i'}))\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})]$, $E[(1 - \pi(\mathbf{x}_{i'}))\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})^\top]$, $E[(1 - \pi(\mathbf{x}_{i'}))\nu(\mathbf{x}_{i'})]$, and $E[(1 - \pi(\mathbf{x}_{i'}))]$, which appear in Eqs. (4.19). When the recording propensities $\nu(\cdot)$ and $\pi(\cdot)$ are known, the following consistent estimators may be used:

$$\begin{aligned} & \hat{E}[(1 - \pi(\mathbf{x}_{i'}))\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})] \\ = & \frac{\sum_{i'=1}^N (\pi(\mathbf{x}_{i'})^{-1} - 1)\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}, \end{aligned} \quad (4.21)$$

$$\begin{aligned} & \hat{E}[(1 - \pi(\mathbf{x}_{i'}))\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})^\top] \\ = & \frac{\sum_{i'=1}^N (\pi(\mathbf{x}_{i'})^{-1} - 1)\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})^\top}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}, \end{aligned} \quad (4.22)$$

$$\begin{aligned} & \hat{E}[(1 - \pi(\mathbf{x}_{i'}))\nu(\mathbf{x}_{i'})] \\ = & \frac{\sum_{i'=1}^N (\pi(\mathbf{x}_{i'})^{-1} - 1)\nu(\mathbf{x}_{i'})}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}, \end{aligned} \quad (4.23)$$

$$\begin{aligned} & \hat{E}[(1 - \pi(\mathbf{x}_{i'}))] \\ = & \frac{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1} - N}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}. \end{aligned} \quad (4.24)$$

However these estimators may generate a heavy the computational burden. Indeed, in

²We would have the smallest asymptotic estimator variance if the Δ_{ij} 's were mutually independent.

Eq. (3.19), the use of these estimators require $O(N)$ computations for each pair (i, j) with a positive τ_{ij} and a total of $O(N^2)$ computations if there are $O(N)$ such pairs. With categorical covariates, this burden may be greatly reduced by first estimating the PMF of the covariates through their empirical distribution in file A. Using this empirical PMF, the above HT estimators may be computed with $O(1)$ computations for each pair (i, j) .

The estimation procedure based on Corollary 8 involve the expectations $E[\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})]$ and $E[\nu(\mathbf{x}_{i'})]$ in Eq. (4.20). The following unbiased HT estimators may be used:

$$\widehat{E}[\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})] = \frac{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1} \nu(\mathbf{x}_{i'}) \mu(\mathbf{x}_{i'}; \boldsymbol{\theta})}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}, \quad (4.25)$$

$$\widehat{E}[\nu(\mathbf{x}_{i'})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})\mu(\mathbf{x}_{i'}; \boldsymbol{\theta})^\top] = \frac{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1} \nu(\mathbf{x}_{i'}) \mu(\mathbf{x}_{i'}; \boldsymbol{\theta}) \mu(\mathbf{x}_{i'}; \boldsymbol{\theta})^\top}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}, \quad (4.26)$$

$$\widehat{E}[\nu(\mathbf{x}_{i'})] = \frac{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1} \nu(\mathbf{x}_{i'})}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}. \quad (4.27)$$

As before, the computation of these estimators is easier with categorical covariates. We next illustrate the above changes by revisiting the linear and logistic regression examples.

Linear regression example: As in Section 3.5.1, we consider the homoschedastic linear model with scalar response y_i such that $E[y_i | \mathbf{x}_i] = \mathbf{x}_i^\top \boldsymbol{\beta}$ and $\text{var}(y_i | \mathbf{x}_i) = \sigma^2$. For the procedure based on Corollary 5, Δ_{ij} , the LSE and the WLSE are still given by Equations (3.21), (3.23) and (3.24), respectively. However, $\mathbf{w}_{ij}$ and σ_{ij}^2 are now

computed as

$$\mathbf{w}_{ij} = \frac{\left[q_{ij} \nu(\mathbf{x}_i) \mathbf{x}_i + (1 - q_{ij}) \left(\frac{\sum_{i' \in s - \{i\}} \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}{\left[q_{ij} \nu(\mathbf{x}_i) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \nu(\mathbf{x}_{i'})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'})]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}, \quad (4.28)$$

$$\sigma_{ij}^2 = \sigma^2 + \frac{\left[q_{ij} \nu(\mathbf{x}_i) (\mathbf{x}_i^\top \boldsymbol{\beta})^2 + (1 - q_{ij}) \left(\frac{\sum_{i' \in s - \{i\}} \nu(\mathbf{x}_{i'}) (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}{\left[q_{ij} \nu(\mathbf{x}_i) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \nu(\mathbf{x}_{i'})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'})]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]} - (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2. \quad (4.29)$$

When the recording propensities are given, the nuisance parameters are $\boldsymbol{\psi}$, σ^2 , $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}]$, $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top]$, $E[(1 - \pi(\mathbf{x}_{i'}))]$, $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'})]$. When $\boldsymbol{\beta}$ and all the other nuisance parameters are known, a

consistent estimator of σ^2 is

$$\hat{\sigma}^2 = \frac{\max \left(0, \sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \tau_{ij} \times \left[\begin{aligned} & q_{ij} \nu(\mathbf{x}_i) (\mathbf{x}_i^\top \boldsymbol{\beta})^2 - \\ & (1 - q_{ij}) \left(\frac{\sum_{i' \in s - \{i\}} \nu(\mathbf{x}_{i'}) (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2}{N_h - 1} + \right. \right. \\ & \left. \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right] \right.}{\sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \tau_{ij}} + (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2 \left. \right) \quad (4.30) \end{aligned}$$

The nuisance parameters that do not involve $\boldsymbol{\beta}$ and σ^2 include $\boldsymbol{\psi}$, $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}]$, $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top]$, $E[(1 - \pi(\mathbf{x}_{i'}))]$, and $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'})]$. The expectations $E[(1 - \pi(\mathbf{x}_{i'}))]$, and $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'})]$ may be estimated based on Eqs. (4.23) and (4.24). The estimators for $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}]$ and $E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top]$ may be based on

the following equations, which complement Eqs. (4.21) and (4.22), respectively:

$$\widehat{E}[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}] = \frac{\sum_{i'=1}^N (\pi(\mathbf{x}_{i'})^{-1} - 1) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}, \quad (4.31)$$

$$\widehat{E}[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top] = \frac{\sum_{i'=1}^N (\pi(\mathbf{x}_{i'})^{-1} - 1) \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}. \quad (4.32)$$

Overall, the computation of these nuisance parameters represent $O(1)$ computations per record-pair instead of $O(N)$ computations for the general regression problem. These estimated nuisance parameters (including $\widehat{\boldsymbol{\psi}}$) may be used as plug-in parameters to compute the LSE of $\boldsymbol{\beta}$, $\widehat{\sigma}^2$ and the WLSE of $\widehat{\boldsymbol{\beta}}$, in this sequence.

For the procedure based on Corollary 8, $\mathbf{w}_{ij}$ and σ_{ij}^2 are computed as

$$\mathbf{w}_{ij} = \frac{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) \mathbf{x}_i + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}] \right)}{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'})] \right)}, \quad (4.33)$$

$$\sigma_{ij}^2 = \sigma^2 + \frac{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) (\mathbf{x}_i^\top \boldsymbol{\beta})^2 + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'}) (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2] \right)}{\left(E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'})] \right)} - (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2. \quad (4.34)$$

Besides σ^2 and $\boldsymbol{\psi}$, the nuisance parameters are $E[\nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}]$, $E[\nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top]$ and $E[\nu(\mathbf{x}_{i'})]$. When all the other parameters are known, a consistent estimator of σ^2 is

$$\hat{\sigma}^2 = \frac{\max \left(0, \sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \tau_{ij} \times \left(\begin{aligned} & E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) (\mathbf{x}_i^\top \boldsymbol{\beta})^2 + \\ & (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'}) (\mathbf{x}_{i'}^\top \boldsymbol{\beta})^2] \end{aligned} \right) \right.}{\sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \tau_{ij}} + (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2 \left. \right) + (\mathbf{w}_{ij}^\top \boldsymbol{\beta})^2 \left(\begin{aligned} & E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) + \\ & (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'})] \end{aligned} \right) \right).$$
(4.35)

The parameter $E[\nu(\mathbf{x}_{i'})]$ may be estimated according to Eq.(4.27), while estimators for $E[\nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}]$ and $E[\nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top]$ may be based on the following equations:

$$\hat{E}[\nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}] = \frac{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1} \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'}}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}, \quad (4.36)$$

$$\hat{E}[\nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top] = \frac{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1} \nu(\mathbf{x}_{i'}) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}. \quad (4.37)$$

All these estimators may be computed with $O(N)$ computations per pair. We next consider the logistic regression example.

Logistic regression example: For the procedure based on Corollary 5, $E[z_j | O_{ij}]$ is

$$E[z_j | O_{ij}] = \frac{\left[q_{ij} \nu(\mathbf{x}_i) \mu_i + (1 - q_{ij}) \left(\frac{\sum_{i' \in s - \{i\}} \nu(\mathbf{x}_{i'}) \mu_{i'}}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'}) \mu_{i'}]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}{\left[q_{ij} \nu(\mathbf{x}_i) + (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \nu(\mathbf{x}_{i'})}{N_h - 1} + \frac{N_h - |s_h|}{N_h - 1} \frac{E[(1 - \pi(\mathbf{x}_{i'})) \nu(\mathbf{x}_{i'})]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right) \right]}, \quad (4.38)$$

while σ_{ij}^2 is still given by Eq. (3.35). When the covariates are categorical and the recording propensities are given, the nuisance parameters include the mixture parameters and the PMF of the covariates, which may be estimated by the empirical distribution of the covariates. This empirical PMF may be used to estimate the expectations that are required in Eq. (4.38) based on Eqs. (4.21)-(4.24). These observations also apply to the procedure based on Corollary 8, where $E[z_j | O_{ij}]$ is

$$E[z_j | O_{ij}] = \frac{\left[E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) \mu_i + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'}) \mu_{i'}] \right]}{\left[E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \nu(\mathbf{x}_i) + (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E[\nu(\mathbf{x}_{i'})] \right]}. \quad (4.39)$$

4.5.2 Maximum composite likelihood

The proposed MCLE remains the solution of Eq. (3.39) where $f_{ij}(\mathbf{z}_j | O_{ij})$ is given by Eq. (4.10) in Corollary 7, or Eq. 4.18 in Corollary 10. The different expectations that appear in Eq. (4.10) may involve additional nuisance parameters. They may be

estimated without bias using the estimators

$$\begin{aligned} & \widehat{E} [(1 - \pi(\mathbf{x}_{i'})) P(j(i') \in B_h | \mathbf{x}_{i'}) f_{y|x \cap B}(\boldsymbol{\zeta} | \mathbf{x}_{i'})] \\ &= \frac{\sum_{i'=1}^N (\pi(\mathbf{x}_{i'})^{-1} - 1) P(j(i') \in B_h | \mathbf{x}_{i'}) f_{y|x \cap B}(\boldsymbol{\zeta} | \mathbf{x}_{i'})}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}, \end{aligned} \quad (4.40)$$

$$\begin{aligned} & \widehat{E} [(1 - \pi(\mathbf{x}_{i'})) P(j(i') \in B_h | \mathbf{x}_{i'})] \\ &= \frac{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1} (1 - \pi(\mathbf{x}_{i'})) P(j(i') \in B_h | \mathbf{x}_{i'})}{\sum_{i'=1}^N \pi(\mathbf{x}_{i'})^{-1}}. \end{aligned} \quad (4.41)$$

The estimator for $E[(1 - \pi(\mathbf{x}_{i'}))]$ is given by Eq. (4.24). We next revisit the survival model example of Section 3.5.2.

Survival model example: As before we have a finite population of individuals and their survival times according to a PHM with exponential survival times. However, the example is slightly modified to correspond to actual mortality data as recorded in the Canadian Mortality Database [2]. We now consider that file B only records the events that have occurred by the end of the follow-up at time T . For each such event, the *uncensored* survival time t_i (of individual i) is recorded. However, file B does not contain any censored time. Indeed, a given vintage of the mortality file only contains the deaths that have occurred by the corresponding reference date (e.g. December 31, 2011), and no information about Canadians that are still alive by that date. Since a survival time t_i is recorded in file B only if $t_i < T$, the survival times are missing in file B according to a *nonignorable* missing data mechanism. Let $f_{t|x}(\cdot | \cdot)$ denote the conditional PDF of the survival time given the covariates. It is

$$f_{t|x}(t | \mathbf{x}_i) = e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \exp\left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} t\right). \quad (4.42)$$

Let $F_{t|x}(\cdot|\cdot)$ denote the corresponding CDF. It is

$$F_{t|x}(t|\mathbf{x}_i) = 1 - \exp\left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} t\right). \quad (4.43)$$

For the estimation procedure based on Corollary 7 and $\zeta < T$, we have

$$\begin{aligned} P(j(i) \in B_h | \mathbf{x}_i) f_{t|x \cap B}(\zeta | \mathbf{x}_i) &= f_{t|x}(\zeta | \mathbf{x}_i) \\ &= e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \exp\left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \zeta\right), \end{aligned} \quad (4.44)$$

$$\begin{aligned} P(j(i) \in B_h | \mathbf{x}_i) &= P(t_i \leq T | \mathbf{x}_i) \\ &= F_{t|x}(T | \mathbf{x}_i) \\ &= 1 - \exp\left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T\right). \end{aligned} \quad (4.45)$$

Then, for $z \leq T$, we have

$$f_{ij}(z | O_{ij}) = \frac{h_{ij}(\boldsymbol{\beta}; z)}{H_{ij}(\boldsymbol{\beta}; T)}, \quad (4.46)$$

where $h_{ij}(\boldsymbol{\beta}; z)$ and $H_{ij}(\boldsymbol{\beta}; T)$ are

$$\begin{aligned}
 h_{ij}(\boldsymbol{\beta}; z) &= q_{ij} e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \exp\left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z\right) + \\
 &\quad (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \exp\left(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z\right)}{N_h - 1} + \right. \\
 &\quad \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E\left[(1 - \pi(\mathbf{x}_{i'})) e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \exp\left(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z\right)\right]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right), \quad (4.47)
 \end{aligned}$$

$$\begin{aligned}
 H_{ij}(\boldsymbol{\beta}; T) &= q_{ij} \left(1 - \exp\left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T\right)\right) + \\
 &\quad (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \left(1 - \exp\left(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T\right)\right)}{N_h - 1} + \right. \\
 &\quad \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E\left[(1 - \pi(\mathbf{x}_{i'})) \left(1 - \exp\left(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T\right)\right)\right]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right). \quad (4.48)
 \end{aligned}$$

As before, the MCLE satisfies Eqs. (3.42) and (3.43) with

$$\frac{\partial}{\partial \boldsymbol{\beta}^\top} \log f_{ij}(z | O_{ij}) = \frac{1}{h_{ij}(\boldsymbol{\beta}; z)} \frac{\partial}{\partial \boldsymbol{\beta}^\top} h_{ij}(\boldsymbol{\beta}; z) - \frac{1}{H_{ij}(\boldsymbol{\beta}; T)} \frac{\partial}{\partial \boldsymbol{\beta}^\top} H_{ij}(\boldsymbol{\beta}; T), \quad (4.49)$$

where

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\beta}^\top} h_{ij}(\boldsymbol{\beta}; z) &= q_{ij} \left(1 - ze^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right) \exp \left(\mathbf{x}_i^\top \boldsymbol{\beta} - e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z \right) \mathbf{x}_i^\top + \\
&\quad (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \left(1 - ze^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp \left(\mathbf{x}_{i'}^\top \boldsymbol{\beta} - e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z \right) \mathbf{x}_{i'}^\top}{N_h - 1} + \right. \\
&\quad \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E \left[(1 - \pi(\mathbf{x}_{i'})) \left(1 - ze^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp \left(\mathbf{x}_{i'}^\top \boldsymbol{\beta} - e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z \right) \mathbf{x}_{i'}^\top \right]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right), \tag{4.50}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\beta}^\top} H_{ij}(\boldsymbol{\beta}; T) &= -q_{ij} \left(-Te^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right) \exp \left(-Te^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right) \mathbf{x}_i^\top - \\
&\quad (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \left(-Te^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp \left(-Te^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \mathbf{x}_{i'}^\top}{N_h - 1} + \right. \\
&\quad \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E \left[(1 - \pi(\mathbf{x}_{i'})) \left(-Te^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp \left(-Te^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \mathbf{x}_{i'}^\top \right]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right). \tag{4.51}
\end{aligned}$$

The second-order derivative of the composite log-likelihood is still given by Eq. (3.46)

with

$$\begin{aligned}
\frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} \log f_{ij}(z | O_{ij}) &= \frac{1}{h_{ij}(\boldsymbol{\beta}; z)} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} h_{ij}(\boldsymbol{\beta}; z) - \\
&\quad \frac{1}{h_{ij}(\boldsymbol{\beta}; z)^2} \left(\frac{\partial}{\partial \boldsymbol{\beta}^\top} h_{ij}(\boldsymbol{\beta}; z) \right) \left(\frac{\partial}{\partial \boldsymbol{\beta}^\top} h_{ij}(\boldsymbol{\beta}; z) \right)^\top - \\
&\quad \frac{1}{H_{ij}(\boldsymbol{\beta}; T)} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} H_{ij}(\boldsymbol{\beta}; T) + \\
&\quad \frac{1}{H_{ij}(\boldsymbol{\beta}; T)^2} \left(\frac{\partial}{\partial \boldsymbol{\beta}^\top} H_{ij}(\boldsymbol{\beta}; T) \right) \left(\frac{\partial}{\partial \boldsymbol{\beta}^\top} H_{ij}(\boldsymbol{\beta}; T) \right)^\top, \tag{4.52}
\end{aligned}$$

where

$$\begin{aligned} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} h_{ij}(\boldsymbol{\beta}; z) = & \\ & q_{ij} \left(\left(1 - z e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right)^2 - z e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right) \exp \left(\mathbf{x}_i^\top \boldsymbol{\beta} - e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z \right) + \\ & (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \left(\left(1 - z e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)^2 - z e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp \left(\mathbf{x}_{i'}^\top \boldsymbol{\beta} - e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z \right)}{N_h - 1} + \right. \\ & \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E \left[(1 - \pi(\mathbf{x}_{i'})) \left(\left(1 - z e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)^2 - z e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp \left(\mathbf{x}_{i'}^\top \boldsymbol{\beta} - e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z \right) \right]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right). \end{aligned} \quad (4.53)$$

$$\begin{aligned} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} H_{ij}(\boldsymbol{\beta}; T) = & \\ & -q_{ij} \left(\left(-T e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right)^2 - T e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right) \exp \left(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} T \right) \mathbf{x}_i \mathbf{x}_i^\top - \\ & (1 - q_{ij}) \left(\frac{\sum_{i' \in s_h - \{i\}} \left(\left(-T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)^2 - T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp \left(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T \right) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top}{N_h - 1} + \right. \\ & \left. \frac{N_h - |s_h|}{N_h - 1} \frac{E \left[(1 - \pi(\mathbf{x}_{i'})) \left(\left(-T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)^2 - T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp \left(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} T \right) \mathbf{x}_{i'} \mathbf{x}_{i'}^\top \right]}{E[(1 - \pi(\mathbf{x}_{i'}))]} \right). \end{aligned} \quad (4.54)$$

The MCLE may be computed using the iterative Newton-Raphson procedure according to Eq. 3.45.

For the estimation procedure based on Corollary 10, write $f_{ij}(z | O_{ij})$ as in Eq. (4.46),

where $h_{ij}(\boldsymbol{\beta}; z)$ and $H_{ij}(\boldsymbol{\beta}; T)$ are

$$\begin{aligned} h_{ij}(\boldsymbol{\beta}; z) &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \exp(-e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z) + \\ &\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E \left[e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \exp(-e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z) \right], \end{aligned} \quad (4.55)$$

$$\begin{aligned} H_{ij}(\boldsymbol{\beta}; T) &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \left(1 - \exp(-T e^{\mathbf{x}_i^\top \boldsymbol{\beta}}) \right) + \\ &\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E \left[\left(1 - \exp(-T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}}) \right) \right]. \end{aligned} \quad (4.56)$$

The first-order derivative of the log-likelihood is based on Eq. (4.49) with

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\beta}^\top} h_{ij}(\boldsymbol{\beta}; z) &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \left(1 - z e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right) \exp(\mathbf{x}_i^\top \boldsymbol{\beta} - e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z) \mathbf{x}_i^\top + \\ &\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E \left[\left(1 - z e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp(\mathbf{x}_{i'}^\top \boldsymbol{\beta} - e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z) \mathbf{x}_{i'}^\top \right] \end{aligned} \quad (4.57)$$

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\beta}^\top} H_{ij}(\boldsymbol{\beta}; T) &= -E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \left(-T e^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right) \exp(-T e^{\mathbf{x}_i^\top \boldsymbol{\beta}}) \mathbf{x}_i^\top - \\ &\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E \left[\left(-T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right) \exp(-T e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}}) \mathbf{x}_{i'}^\top \right] \end{aligned} \quad (4.58)$$

The second-order derivative of the log-likelihood is based on Eq. (4.52) with

$$\begin{aligned} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} h_{ij}(\boldsymbol{\beta}; z) &= E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \frac{\left((1 - ze^{\mathbf{x}_i^\top \boldsymbol{\beta}})^2 - ze^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right)}{\exp(-\mathbf{x}_i^\top \boldsymbol{\beta} + e^{\mathbf{x}_i^\top \boldsymbol{\beta}} z)} \mathbf{x}_i \mathbf{x}_i^\top + \\ &\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E \left[\frac{\left((1 - ze^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}})^2 - ze^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)}{\exp(-\mathbf{x}_{i'}^\top \boldsymbol{\beta} + e^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} z)} \mathbf{x}_{i'} \mathbf{x}_{i'}^\top \right] \end{aligned} \quad (4.59)$$

$$\begin{aligned} \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} H_{ij}(\boldsymbol{\beta}; T) &= -E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h] \frac{\left((-Te^{\mathbf{x}_i^\top \boldsymbol{\beta}})^2 - Te^{\mathbf{x}_i^\top \boldsymbol{\beta}} \right)}{\exp(Te^{\mathbf{x}_i^\top \boldsymbol{\beta}})} \mathbf{x}_i \mathbf{x}_i^\top - \\ &\quad (1 - E[q_{ij} | \mathbf{x}_i, \gamma_{ij}, i \in A_h]) E \left[\frac{\left((-Te^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}})^2 - Te^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}} \right)}{\exp(Te^{\mathbf{x}_{i'}^\top \boldsymbol{\beta}})} \mathbf{x}_{i'} \mathbf{x}_{i'}^\top \right] \end{aligned} \quad (4.60)$$

4.6 Large sample theory

We may examine the consistency and asymptotic normality of the proposed estimators by applying the arguments of Section 3.6 using the expressions for Δ_{ij} and $f_{ij}(\cdot | O_{ij})$ given in Section 4.5.

4.7 Simulation study

The Monte-Carlo simulation study of Section 3.7 is enhanced with missing data mechanisms. The following paragraphs are organized as follows. Section 4.7.1 describes the simulation setup. Section 4.7.2 presents the results for the linear model. Section 4.7.3 presents the results for the logistic model. Section 4.7.4 presents the results

for the survival model. Section 4.7.5 presents the conclusions.

4.7.1 General setup

The setup of Section 3.7.1 is enhanced with missing data mechanisms. For the linear model and the logistic model, the response y_i is missing in register B according to a logistic model with covariate x_i and coefficients $\beta' = [-2 \ 1]^\top$, while the covariate x_i is missing in register A according to another logistic model with covariate x_i and coefficients $\beta'' = [-2 \ 1]^\top$. The parameters of the missing data mechanism are considered known.

4.7.2 Linear model

The same parameters (as in Section 3.7.2) are used for the conditional distribution of $\mathbf{y}_i$ given $\mathbf{x}_i$. Four estimators are evaluated including the naive estimator, the complete data estimator, and two WLS pairwise estimators according to the linear regression example of Section 4.5.1. The naive estimator and the complete data estimator are still the solutions of Eq. (3.65) and Eq. (3.66) respectively. Note that in these equations, A_h and B_h denote the samples of selected records that are included in file A and file B, in block h .

4.7.3 Logistic model

The same parameters (as in Section 3.7.3) are used for the conditional distribution of $\mathbf{y}_i$ given $\mathbf{x}_i$. Four estimators are evaluated including the naive estimator, the complete data estimator, and two WLS pairwise estimators according to the logistic regression

Table 4.1: Linear model when linking two samples with $N_h = 2$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	-3.479	0.004889	0.005143
	PW1	-2.02	0.004922	0.004974
	PW2	-2.016	0.004849	0.004902
	Complete	-1.263	0.002796	0.002808
β_1	Naive	-5.674	0.006667	0.00982
	PW1	-2.294	0.007249	0.007703
	PW2	-2.323	0.007192	0.007659
	Complete	-1.011	0.004609	0.004665

Table 4.2: Linear model when linking two samples with $N_h = 2$ and a CMP threshold of 0.0.

Parameter	Method	Bias (%)	Variance	MSE
β_0	PW1	0.705	0.003047	0.003029
	PW2	1.245	0.003161	0.003168
β_1	PW1	0.049	0.003693	0.003657
	PW2	0.26	0.005242	0.005196

Table 4.3: Linear model when linking two samples with $N_h = 8$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	-3.433	0.001622	0.001901
	PW1	-0.551	0.001505	0.001498
	PW2	-0.574	0.001563	0.001556
	Complete	-0.121	0.00065	0.000644
β_1	Naive	-6.649	0.002736	0.007129
	PW1	-0.515	0.002745	0.002744
	PW2	-0.551	0.002816	0.002818
	Complete	-0.287	0.000786	0.000786

example in Section 4.5.1. The naive estimator and the complete data estimator are based on Eq. (3.67) and Eq. (3.68) respectively.

4.7.4 Survival model

We consider a different missing data mechanism where only the noncensored survival times are reported in file B, i.e. the file contains no information about the subjects that have not experienced the event by the end of the follow-up. This is closer to the reality when file B is a mortality file such as the Canadian Mortality Database (CMDB). In this case, the responses are not missing at random and B_h only contains the noncensored survival times that are each necessarily smaller than T . See the example of Section 4.5.2 for further details. The naive estimator is based on the

Table 4.4: Logistic model when linking two samples.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	3.811	0.029602	0.029669
	PW1	4.732	0.030067	0.030326
	PW2	4.652	0.030144	0.030383
	Complete	3.215	0.02333	0.023355
β_1	Naive	-8.21	0.095634	0.101417
	PW1	-4.806	0.100836	0.102138
	PW2	-5.053	0.101983	0.103516
	Complete	-3.616	0.069176	0.069792

Table 4.5: Logistic model when linking two samples with $N_h = 2$ and a CMP threshold of 0.2.

Parameter	Method	Bias (%)	Variance	MSE
β_0	PW1	-1.908	0.026487	0.026313
	PW2	-1.8	0.026535	0.02635
β_1	PW1	5.123	0.112712	0.114209
	PW2	5.791	0.106298	0.108589

Table 4.6: Logistic model when linking two samples with $N_h = 8$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	0.35	0.009203	0.009114
	PW1	2.157	0.009626	0.009646
	PW2	2.17	0.009608	0.00963
	Complete	2.988	0.005468	0.005637
β_1	Naive	-5.806	0.049469	0.052346
	PW1	0.424	0.057561	0.057004
	PW2	0.542	0.057566	0.057019
	Complete	1.563	0.023516	0.023525

following equation.

$$\hat{\beta} = \arg \max_{\beta} \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} l_{ij} \log \left(\frac{e^{\mathbf{x}_i^\top \beta} \exp(-e^{\mathbf{x}_i^\top \beta} z_j)}{1 - \exp(-e^{\mathbf{x}_i^\top \beta} T)} \right) \quad (4.61)$$

where $\mathbf{x}_i = [1 \ x_i]^\top$, $l_{ij} = I(q_{ij} \geq q_0)$, where $q_{ij} = P(m_{ij} = 1 | \gamma_{ij})$ and $q_0 = 0.9$. Note that As for the complete data estimator, it is the solution of the following optimization problem.

$$\hat{\beta} = \arg \max_{\beta} \sum_{h=1}^H \sum_{(i,j) \in A_h \times B_h} m_{ij} \log \left(\frac{e^{\mathbf{x}_i^\top \beta} \exp(-e^{\mathbf{x}_i^\top \beta} z_j)}{1 - \exp(-e^{\mathbf{x}_i^\top \beta} T)} \right) \quad (4.62)$$

The two pairwise estimators are based on the methodology described in Section 4.5.1. For $N_h = 8$, PW1 is still better than PW2 for the same reason as before, i.e. conditioning on all block covariates vs. conditioning on the pair covariates, but the difference is much bigger. The MSEs of the two estimators differ by many orders of

magnitude.

Table 4.7: Survival model when linking two samples.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	2.234	0.022896	0.022792
	PW1	-2.496	0.022546	0.022476
	PW2	-2.175	0.023346	0.023231
	Complete	-0.143	0.018608	0.018422
β_1	Naive	-5.332	0.009754	0.012499
	PW1	-0.061	0.008479	0.008395
	PW2	-0.226	0.008608	0.008527
	Complete	0.211	0.006565	0.006504

Table 4.8: Survival model when linking two samples with $N_h = 2$ and a CMP threshold of 0.0.

Parameter	Method	Bias (%)	Variance	MSE
β_0	PW1	-3.64	0.008302	0.00855
	PW2	0.966	0.015819	0.015684
β_1	PW1	1.784	0.003778	0.004058
	PW2	0.895	0.006007	0.006028

4.7.5 Conclusions

The simulation results show that pairwise estimators, where the conditioning is based on the information from an entire block, tend to be superior to pairwise estimators,

Table 4.9: survival model when linking two samples with $N_h = 8$ and a CMP threshold of 0.9.

Parameter	Method	Bias (%)	Variance	MSE
β_0	Naive	-57.384	10.318838	10.297974
	PW1	-1.399	0.006326	0.006312
	PW2	-8.099	0.133355	0.133661
	Complete	-0.144	0.00256	0.002535
β_1	Naive	7.438	2.578701	2.558447
	PW1	0.167	0.002663	0.00264
	PW2	1.78	0.033379	0.033362
	Complete	-0.081	0.001022	0.001013

where the conditioning is based on the information within a single pair. For the proposed composite likelihood estimators, the gain is possibly large in scenarios where the missing data mechanism for the responses is informative, i.e. nonignorable, as in cohort mortality studies. In practice, implementing PW1 is a challenge because it requires the actual block sizes N_h in the original finite population. However, this information is not directly observed except when file A, the data source for the covariates, is a register. In all other cases, a practical alternative is to use the other pairwise estimators despite the obvious efficiency loss.

Chapter 5

Model-assisted EEs

5.1 Introduction

In the previous two chapters, we have described a model-based methodology that requires no clerical-review but relies on various conditional independence assumptions and the correct specification of the marginal mixture model for a pair. The resulting estimators become inconsistent if these assumptions do not hold or the marginal mixture model is misspecified. The current chapter describes an alternative methodology that overcomes these limitations when a clerical-review sample is available [50]. It is a contribution by the author, who was inspired by his experience with the 2011 census overcoverage study [51,52]. The main idea is to accurately estimate any finite population total of the form $\sum_{i=1}^N \mathbf{g}(\mathbf{y}_i, \mathbf{x}_i; \boldsymbol{\theta}) = \sum_{(i,j)} m_{ij} \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i; \boldsymbol{\theta})$ by using a probability sample of pairs that have a known match status, and a model-based predictor $\hat{m}_{ij}$ of the match status for all the pairs that meet the blocking criteria. In turn, the estimated totals may be used to estimate superpopulation parameters by solving the corresponding unbiased EEs. The proposed estimators are examples of regression estimators, which include generalized regression estimators (GREG) and calibration

estimators that use auxiliary variables. These estimators have been thoroughly studied by Sarndal et al. [53] and by Deville and Sarndal [54]. They are also referred to as model-assisted estimators because they are inspired by some implicit statistical model; typically a linear model relating the auxiliary variables to the variables of interest. In general, these estimators are more efficient than the Horwitz-Thompson estimator when the model holds and the auxiliary variables are strongly correlated with the variable of interest. However, they remain design-consistent regardless of the model validity [53, section 6.7, pp. 239].

The following sections closely follow [50], where the focus is on the estimation of a finite population total. However, minor changes therein include an alignment of the notation with that of previous chapters and the assumption of perfect blocks. Section 5.3 describes the proposed estimators. Section 5.4 discusses the choice of the sampling design for the clerical-review sample. Section 5.5 presents some simulation results. Section 5.6 presents some conclusions.

5.2 Notation

Consider the setting (the linkage of two duplicate-free registers of the the same finite population) and notation of Chapter 3, which we now briefly recall. We have a finite population of N individuals that are distributed across H IID and variable size clusters called blocks, with IID individuals with each block and an homogeneous distribution of the individuals across the blocks. Block h includes N_h individuals, where $N_h \leq C$ for a constant C regardless of H and $N = N_1 + \dots + N_H$. Each individual has a set of attributes including quasi-identifiers, covariates and related responses. These attributes are recorded in two population registers A and B that are linked. The first register (A) contains the block identifier, the quasi-identifiers and the covariates

for each individual, while the second register (B) contains the block identifier, the quasi-identifiers and the responses for the same individual. The recorded block identifier, responses and covariates are error-free but the quasi-identifiers are recorded with errors in each file. Let $\mathbf{x}_i$ denote the covariates in record i from register A , $\mathbf{y}_i$ the corresponding (unobserved) responses and $\mathbf{z}_j$ denote the observed responses in record j from register B . Block h corresponds to records indexed in two known subsets A_h and B_h (of $\{1, \dots, N\}$) in registers A and B respectively. These subsets contain the same number of records (i.e. $|A_h| = |B_h| = N_h$) and are such that each record in A_h has a single matching record in B_h . The collections of subsets $A_1, \dots, A_H$ and $B_1, \dots, B_H$ form two partitions of $\{1, \dots, N\}$. For a record-pair $(i, j) \in A_h \times B_h$, γ_{ij} and m_{ij} denote the related comparison vector and match status indicator respectively. Blocking is based on the recorded block identifier that is assumed error-free. Consequently the blocking criteria are perfect.

The comparison vector γ_{ij} is used for making a prediction $\hat{m}_{ij}$ about the unknown match status m_{ij} . This prediction can take many forms. For example, it can be set to the conditional or posterior match probability $P(m_{ij} = 1 | \gamma_{ij})$ given the comparison vector. It can also be interpreted as the weight-share of the pair (i, j) , with the meaning of the generalized weight share method. See Lavallée [55, chap. 9] for applications of this method to record-linkage. For finite population inference, the goal is estimating a total of the form

$$\mathbf{G} = \sum_{i=1}^N \mathbf{g}(\mathbf{y}_i, \mathbf{x}_i) \quad (5.1)$$

$$= \sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} m_{ij} \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i), \quad (5.2)$$

where $\mathbf{g}(\cdot, \cdot)$ is a known function. The above total may be used to estimate a superpopulation parameter $\boldsymbol{\theta}_0$, which is such that $E[\mathbf{g}(\mathbf{y}_i, \mathbf{x}_i; \boldsymbol{\theta}_0)] = 0$. For example,

$\mathbf{g}(\cdot, \cdot)$ may be the the derivative of the log-likelihood of the conditional distribution of $\mathbf{y}_i$ given $\mathbf{x}_i$, if it has a known parametric form. In this case, the parameter may be estimated through the solution $\hat{\boldsymbol{\theta}}$ of the unbiased EE.

$$\mathbf{G}(\hat{\boldsymbol{\theta}}) = \sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} m_{ij} \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i; \hat{\boldsymbol{\theta}}) = \mathbf{0}. \quad (5.3)$$

Resources for error-free clerical reviews are available to measure the match status. However, they are costly and must be minimized. The clerical sample s has a fixed size. Let π_{ij} denote the corresponding first-order sample inclusion probability for the record-pair (i, j) .

5.3 Model-assisted estimators

The proposed estimators are regression estimators [53, chap. 6] that have the general difference form

$$\hat{\mathbf{G}} = \sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \hat{m}_{ij} \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) + \sum_{(i,j) \in s} \pi_{ij}^{-1} (m_{ij} - \hat{m}_{ij}) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i). \quad (5.4)$$

The above estimator may be viewed as a calibration estimator [54], where the estimated total is calibrated to the corresponding total based on inferred match status. It estimates the total with no sampling error and no bias when the blocking criteria are perfect (i.e., they select all matched pairs) and the predicted match status is error-free, i.e. $\hat{m}_{ij} = m_{ij}$. The estimator is unbiased if the predicted status $\hat{m}_{ij}$ is not based on the the information from the clerical sample. Then

$$E \left[\hat{\mathbf{G}} \mid [(\mathbf{z}_j, \mathbf{x}_i, m_{ij})]_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \right] = \sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} m_{ij} \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i) = \mathbf{G}. \quad (5.5)$$

This is the case if $\hat{m}_{ij}$ is only a function of $\mathbf{z}_j$, $\mathbf{x}_i$ and γ_{ij} . The inferred status may be set to the conditional match probability given these variables, i.e.

$$\hat{m}_{ij} = P(m_{ij} | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}). \quad (5.6)$$

This particular inference strategy would minimize the mean squared error (over the super population) between the predicted total and the actual, among all inference strategies where $\hat{m}_{ij}$ is only a function of $\mathbf{z}_j$, $\mathbf{x}_i$ and γ_{ij} , if the record-pairs were IID. Under a simple random sampling (SRS) design, the resulting estimator would also be more efficient than the Horwitz-Thompson estimator, if the pairs were IID.

The conditional match probability may be estimated under the assumption of IID pairs according to a two component mixture distribution, where the different comparison outcomes and the analytical variables $(\mathbf{x}_i, \mathbf{z}_j)$ are assumed conditionally independent given the match status, where $\tau = 0, 1$. In this case,

$$P(\mathbf{z}_j, \mathbf{x}_i, \gamma_{ij} | m_{ij} = \tau) = P(\mathbf{z}_j, \mathbf{x}_i | m_{ij} = \tau) \prod_{k=1}^K P(\gamma_{ij}^{(k)} | m_{ij} = \tau). \quad (5.7)$$

The parameters $\boldsymbol{\psi}$ of this mixture include the mixing proportion $\lambda = P(m_{ij} = 1)$, the m-probabilities $P(\mathbf{z}_j, \mathbf{x}_i | m_{ij} = 1)$ and $P(\gamma_{ij}^{(k)} | m_{ij} = 1)$, and the u-probabilities $P(\mathbf{z}_j, \mathbf{x}_i | m_{ij} = 0)$ and $P(\gamma_{ij}^{(k)} | m_{ij} = 0)$. They may be estimated with an EM algorithm. See Jaro [19] or Winkler [56] for applications of EM to record-linkage, and Dempster et al. [57] for a general reference on EM. An important feature of this mixture model is the use of $\mathbf{z}_j$ and $\mathbf{x}_i$ as additional linkage variables. It becomes simpler when these variables are highly correlated with the usual linkage variables, because they then bring no new information about the match status, given γ_{ij} . Mathematically, this is expressed by the conditional independence of $(\mathbf{x}_i, \mathbf{z}_j)$ and the match

status m_{ij} given the comparison outcomes, i.e.

$$P(m_{ij} = 1 | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}) = P(m_{ij} = 1 | \gamma_{ij}). \quad (5.8)$$

The prediction strategy may be inefficient if the assumed mixture model does not hold. For example, this problem may occur if the couple $(\mathbf{x}_i, \mathbf{z}_j)$ contains additional information about the match status, but the predictor $P(m_{ij} = 1 | \gamma_{ij})$ is used instead. The estimator is also less efficient if the linkage variables are correlated but their conditional independence is assumed. Let $P(m_{ij} = 1 | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}; \hat{\boldsymbol{\psi}})$ denote a preliminary estimate of the conditional match probability according to the mixture model. This estimate is computed in the E-Step of the EM algorithm without the clerical results. In most cases, this mixture model will estimate the conditional match probability with some bias even if it accounts for some of the interactions among the different variables. To adjust for this bias, the match status may instead be predicted using a linear function $b_0 + b_1 P(m_{ij} = 1 | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}; \hat{\boldsymbol{\psi}})$ of the estimated conditional probability, where the regression coefficients b_0 and b_1 are estimated from the clerical sample. In this case, the inferred match status is computed as

$$\hat{m}_{ij} = \hat{b}_0 + \hat{b}_1 P(m_{ij} = 1 | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}; \hat{\boldsymbol{\psi}}). \quad (5.9)$$

A special case (of estimator given by Eq. (5.4) where the predictor $\hat{m}_{ij}$ is given by Eq. (5.9)) is the ratio estimator

$$\hat{\mathbf{G}} = \frac{\sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} P(m_{ij} = 1 | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}; \hat{\boldsymbol{\psi}}) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)}{\sum_{(i,j) \in s} \pi_{ij}^{-1} P(m_{ij} = 1 | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}; \hat{\boldsymbol{\psi}}) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)} \sum_{(i,j) \in s} \pi_{ij}^{-1} m_{ij} \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i). \quad (5.10)$$

In this case, $\widehat{b}_0 = 0$ and $\widehat{b}_1$ is computed as

$$\widehat{b}_1 = \frac{\sum_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} P(m_{ij} = 1 | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}; \widehat{\boldsymbol{\psi}}) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)}{\sum_{(i,j) \in s} \pi_{ij}^{-1} P(m_{ij} = 1 | \mathbf{z}_j, \mathbf{x}_i, \gamma_{ij}; \widehat{\boldsymbol{\psi}}) \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i)}. \quad (5.11)$$

The estimator can also be written in terms of an adjustment (called *g-weight* in the survey sampling literature) $[\eta_{ij}]_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h}$ to the original sampling weights π_{ij}^{-1} . For the above ratio estimator, we have a uniform adjustment $\eta_{ij} = \widehat{b}_1$ and

$$\widehat{\mathbf{G}} = \sum_{(i,j) \in s} \eta_{ij} \pi_{ij}^{-1} m_{ij} \mathbf{g}(\mathbf{z}_j, \mathbf{x}_i). \quad (5.12)$$

The following model provides the basis for better weighted least squares estimators:

$$E[m_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}] = b_0 + b_1 P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}; \boldsymbol{\psi}), \quad (5.13)$$

$$\text{var}(m_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) \propto P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}; \boldsymbol{\psi}) [1 - P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}; \boldsymbol{\psi})]. \quad (5.14)$$

In this case, the estimated regression coefficients minimize the quadratic function

$$Q(b_0, b_1; \boldsymbol{\psi}) = \sum_{(i,j) \in s} \pi_{ij}^{-1} \frac{[m_{ij} - b_0 - b_1 P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}; \boldsymbol{\psi})]^2}{P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}; \boldsymbol{\psi}) [1 - P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}; \boldsymbol{\psi})]}. \quad (5.15)$$

The resulting estimator may be written in terms of nonuniform g-weights incorporating the inferred match status. This estimator is improved by fine tuning the variance structure with generalized estimating equations [58]. The proposed estimators are no longer unbiased because the clerical review results are used to make inferences about the pairs match status. However, like other regression estimators [53, Result 6.6.1, pp. 235, section, 6.7, pp. 238], they are design-consistent regardless of the validity of

the assumed model.

A logistic model is another natural choice for predicting the dichotomous match status m_{ij} . In this case we may use the predictor $\hat{m}_{ij}$ such that

$$\text{logit}(\hat{m}_{ij}) = \hat{b}_0 + \hat{b}_1 \text{logit}(P(m_{ij} = 1 | x_i, y_j, \gamma_{ij})),$$

where

$$(\hat{b}_0, \hat{b}_1) = \arg \min_{b_0, b_1} \sum_{(i,j) \in s} \pi_{ij}^{-1} \frac{[m_{ij} - \mu_{ij}(b_0, b_1)]^2}{\mu_{ij}(b_0, b_1) [1 - \mu_{ij}(b_0, b_1)]},$$

and $\mu_{ij}(b_0, b_1)$ is such that $\text{logit}(\mu_{ij}(b_0, b_1)) = b_0 + b_1 \text{logit}(P(m_{ij} = 1 | x_i, y_j, \gamma_{ij}))$.

Another choice is to draw the predictor $\hat{m}_{ij}$ from a Bernoulli distribution with parameter μ_{ij} .

5.4 Sampling design

Model-based stratified sampling has been used to approximately minimize the variance of regression estimators [53]. In this design, the strata are defined by the variance of the error in the assumed linear model. This strategy also applies to the current context where a single total is estimated, i.e. G and $g(.,.)$ are scalar. The design-based variance $\text{var} \left(\hat{G} \middle| \{(\mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}, m_{ij})\}_{(i,j) \in \bigcup_{h=1}^H A_h \times B_h} \right)$ is approximately minimized by a Neyman allocation where the pairs are stratified according to $\text{var}(e_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij})$,

where $e_{ij} = (m_{ij} - \hat{m}_{ij}) g(\mathbf{z}_j, \mathbf{x}_i)$. This conditional variance is given by

$$\begin{aligned}
 \text{var}(e_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) &= \text{var}(e_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) \\
 &= g(\mathbf{z}_j, \mathbf{x}_i)^2 \text{var}(m_{ij} - \hat{m}_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) \\
 &= g(\mathbf{z}_j, \mathbf{x}_i)^2 \left(\text{var}(m_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) + \right. \\
 &\quad \left. [\hat{m}_{ij} - P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij})]^2 \right) \\
 &= g(\mathbf{z}_j, \mathbf{x}_i)^2 \left(P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) [1 - P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij})] + \right. \\
 &\quad \left. [\hat{m}_{ij} - P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij})]^2 \right).
 \end{aligned}$$

When $\hat{m}_{ij} = P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij})$, we have

$$\text{var}(e_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) = g(\mathbf{z}_j, \mathbf{x}_i)^2 \hat{m}_{ij} (1 - \hat{m}_{ij})$$

We may use a Neyman allocation based on the above expression. To get some insight, suppose that $\mathbf{x}_i$ and $\mathbf{z}_j$ are categorical and take a few distinct values, and suppose that the pairs are stratified based on γ_{ij} and $(\mathbf{x}_i, \mathbf{z}_j)$. Note that by design, in such a stratum, the pairs have the same $g_{ij} = g(\mathbf{z}_j, \mathbf{x}_i)$ value and an identical conditional match probability $P(m_{ij} = 1 | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) = p_{ij}$. Thus, they are identically distributed according to $g_{ij} \text{Bernoulli}(p_{ij})$. If these pairs were independent, the variance of the errors e_{ij} would be well approximated by the common variance $\text{var}(e_{ij} | \mathbf{x}_i, \mathbf{z}_j, \gamma_{ij}) = g_{ij}^2 p_{ij} (1 - p_{ij})$, based on the law of large numbers (LLN). In the corresponding Neyman allocation, the sample size is proportional to the stratum variance. An estimator with the same minimum variance is obtained via a Neyman allocation, where the stratum variance is based on $g_{ij}^2 \hat{m}_{ij} (1 - \hat{m}_{ij})$, the estimated conditional error variance.

5.5 Simulations

5.5.1 Setup

The proposed estimators are evaluated in six scenarios that differ according to

- the discriminating power of the linkage variables,
- the sample size,
- the statistical distribution of linkage errors,
- the rate of clerical errors, and
- the correlation among the pairs.

All the scenarios consider a one-to-one linkage between two registers. In each register the records are partitioned into perfect blocks of equal size. Consequently two matched records always fall within the same block. The different scenarios account for different features of practical linkages. They emulate the process by which administrative records may be generated from a finite population of individuals, including correlations within blocks. Each individual has seven dichotomous attributes that are generated according to a conditionally independent multinomial mixture with two components, and IID attributes in each component. These attributes are recorded in two files, such that for each individual, the recorded errors for the different attributes and files are conditionally independent and identically distributed with probability α , given the individual's attribute. This setup implies the conditional independence of linkage variables if the individual attributes are generated according to a mixture, where the two components have the same distribution. This distribution is given by

the following transition probabilities:

$$\begin{aligned} P\left(c_i^{(k)}, c_j^{(k)} \mid \zeta_i^{(k)}, M_{ij} = 1\right) &= P\left(c_i^{(k)} \mid \zeta_i^{(k)}\right) P\left(c_j^{(k)} \mid \zeta_i^{(k)}\right) \\ P\left(c_i^{(k)} \mid \zeta_i^{(k)}\right) &= (1 - \alpha)I\left((c_i^{(k)} = \zeta_i^{(k)})\right) + \alpha I\left((c_i^{(k)} \neq \zeta_i^{(k)})\right), \end{aligned}$$

where α is the probability of a recording error.

In the above expressions, a_{ik} is the k -th linkage variable for record i in register A, $\zeta_i^{(k)}$ is the latent true (i.e. free of recording errors) value of the variable for the associated individual, with $c_j^{(k)}$ and $\zeta_j^{(k)}$ denoting the corresponding variables in register B. Note that, by definition $\zeta_i^{(k)} = \zeta_j^{(k)}$ in a matched pair (i, j) . For each record i , the latent variables $\zeta_i^{(k)}$ are IID.

Let a_{ik} denote the recorded value for attribute k and individual i in the first file. In a similar manner, let b_{jk} denote the recorded value for attribute k and individual j in the second file. The comparison outcomes are based on exact comparisons with $\gamma_{ij}^{(k)} = I(a_{ik} = b_{jk})$.

The variables of interest x_i and z_j are also dichotomous and mutually independent of the linkage variables in each register, and each matched pair. The files are linked to study the joint distribution of these two variables, i.e. to estimate the frequencies of the different cells in a two-way contingency table. In this case, $g(z_j, x_i) = I((x_i, z_j) = (x, z))$, where $x, z = 0, 1$.

Different IID samples are drawn using one of two designs. For each resulting sample, three estimators are computed for the number of matched pairs in each cell of the two-way contingency table. They include the H-T estimator, a second model-assisted estimator (hereafter simply referred to as 2nd estimator) using the inference $\hat{m}_{ij} = P\left(m_{ij} = 1 \mid x_i, z_j, \gamma_{ij}; \hat{\psi}\right)$ and a third estimator (hereafter simply referred to as 3rd estimator) using the inference $\hat{m}_{ij} = \hat{b}_0 + \hat{b}_1 P\left(m_{ij} = 1 \mid x_i, z_j, \gamma_{ij}; \hat{\psi}\right)$.

The first sample design is stratified according to the x-y value pairs. In each stratum, a fixed size SRS sample is drawn. The second sample design is also stratified based on the x-y value pairs, but it uses substrata, which are based on the conditional variance of the prediction error. Each x-y stratum has the same number of substrata but the boundaries are selected to obtain nearly equal substrata sizes, after the pairs are sorted according to their conditional variance in each stratum. Consequently substrata boundaries may differ from an x-y stratum to the next. The same x-y stratum sample size is used as in the first design. However in the second sample design, this sample size is allocated optimally among the substrata using a Neyman allocation, where the estimated variance of a substratum is estimated as the mean conditional error variance over all the corresponding pairs. A substratum sample allocation is further constrained to have at least two units and not to exceed the substratum size.

Scenario 1 is the baseline scenario. It evaluates the two model-assisted estimators in the best case, with the correct model for the comparison outcomes. This situation maximizes their relative advantage over the nave H-T estimator. Scenarios 2 through 5 are built after Scenario 1, i.e. with correlated pairs. However they each incorporate a slight modification. Scenario 2 considers linkage variables with more typographical errors and hence less discriminating power than in Scenario 1. Scenario 3 considers a smaller (1,000 pairs instead of 4,000 pairs) clerical-review sample. Scenario 4 considers linkage variables that are not conditionally independent, by letting the two mixture components have different distributions when generating the individuals' attributes by correlating the latent variables $\zeta_i^{(k)}$. This correlation is produced by generating the $\zeta_i^{(k)}$ s according to a mixture model with conditional independence based on a binary latent class ξ_i . However the estimated conditional match probability $P\left(m_{ij} = 1 \mid x_i, z_j, \gamma_{ij}; \hat{\psi}\right)$ is estimated under the assumption of conditional

independence among all linkage variables. Scenario 5 considers clerical-errors. Scenario 6 considers agreement frequencies for variables such as names and birthdate that have been used for linking high quality person files. The specific frequencies are based on an example provided by [1, Table 5.1]. Unlike the other scenarios, Scenario 6 considers pairs with IID and conditionally independent comparison vectors.

The simulation parameters are as follows. All scenarios are based on $N = 10,000$ individuals, 1,000 blocks, a block size of 10, $K = 7$ linkage variables, $P(x = 1) = 0.5$, $P(y = 1|x = 0) = 0.4$, $P(y = 1|x = 1) = 0.7$, 10 substrata per x-y stratum, 100 E-M iteration and 100 repetitions. The x-y stratum sample size is set to 1,000 for all scenarios except for scenario 3 (smaller clerical sample), where it is set to 100. The conditional agreement probabilities are uniform across the linkage variables in scenarios 1 through 5. However, they vary across these scenarios. For scenarios 1 and 3 through 5, the conditional probability of agreement is 0.98 for a matched pair and 0.5 for an unmatched pair. For scenario 2, these conditional probabilities are respectively 0.82 and 0.5. For scenario 6, the conditional agreement probabilities are given in Table 5.1. The remaining parameters only apply to scenarios 1 through 5 and are set as follows. The parameter α is set to 0.1 for scenarios 1 through 5. In all the scenarios, except scenario 4, each individual's attribute is uniformly distributed over $\{0, 1\}$ in each mixture component. In scenario 4, an attribute is set to 1 with probability 0.3 in the first component and with probability 0.7 in the second component. For all the scenarios, except scenario 2, the conditional probability of a typographical error (denoted by α) is set to 0.01, for each attribute and each file. In scenario 2, this parameter is instead set to 0.1.

5.5.2 Results

Tables 5.2 and 5.3 show the average bias and CV of the estimated count for cell (0,0) for the different estimators and scenarios. The results for the other cells are not shown because they are similar to those of cell (0,0). As for the third estimator, the corresponding results are not shown because they are similar to those of the second estimator. Figures 5.1 to 5.6 show the box plots for the different scenarios and estimators.

For Scenario 1 (our baseline), all three estimators have a very small relative bias, with no clear advantage for the H-T estimator under either sampling design. However the model-assisted estimator halves the CV of the H-T estimator, under the first sampling design. The gain in precision becomes negligible under the second sampling design. This is expected because the model information is already exploited through the stratification, which also benefits the H-T estimator.

The results for Scenario 2 show a worse performance for the model-assisted estimator, when the linkage variables are less discriminating. Indeed, the corresponding absolute relative bias is larger than that of the H-T estimator, under either sampling design. As for the expected gain in precision under the first sampling design, it is dramatically smaller than in Scenario 1. Under the second design, the gain is negligible.

The results for Scenario 3 show the same trends as in Scenario 1, with similar gains in precision for the model-assisted estimator. Intuitively the use of a model partially makes up for the reduced sample size.

For Scenario 4, where the model is misspecified, both the H-T estimator and the model-assisted estimator have a small relative bias, under either design. For the model-assisted estimator, the gain in precision is slightly reduced compared to Scenario 1.

In Scenario 5, with clerical-errors, Table 2 shows that the relative bias of all the

estimators is significantly increased compared to Scenario 1. However, under the first sampling design, the model-assisted estimators offer a significant advantage over the HT estimator, even if this advantage is smaller than in Scenario 1. Under the second design, this gain in precision vanishes and all the estimators have much less precision than in the first sampling design.

In Scenario 6, the model-assisted estimator greatly outperforms the H-T estimator both regarding the bias and the precision, under either sampling design. The gain in precision is also dramatically larger than in the other scenarios. This is because in Scenario 6, the linkage variables collectively provide much more discrimination than in the previous scenarios. The combination of this discrimination with a correct statistical model produces the observed gains. Overall, the model-assisted estimators offer the best performance when the following three conditions are met:

- i. The linkage variables provide a high discrimination.
- ii. The clerical-reviews are very reliable.
- iii. The assumed statistical model is correct.

Of the above three conditions, the reliability of the clerical-review is the most critical one as it may be expected. The simulation results also shed some light on the choice of the sampling design. In all scenarios without clerical-errors, the precision is much greater under the second sampling design, where the pairs are stratified according to the estimated conditional match probability. This result further underscores the importance of using auxiliary variables that leverage the comparison outcomes. Although this work considers a one-to-one linkage, this assumption does not play a major role in the estimation procedure. Hence the proposed methodology also applies to an incomplete linkage so long as the clerical reviews remain error-free. However the

resulting model-assisted estimators may be less efficient if the unmatchable records greatly differ in distribution from the other records. Then the pairs outcomes are better modeled by a three component mixture including two classes of unmatched pairs. In this case, specifying a good model may be more challenging.

Table 5.1: Agreement frequencies for Scenario 6 based on [1, Table 5.1].

Linkage variable	Agreement probability	
	Matched	Unmatched
Surname	0.965	0.001
First name	0.79	0.009
Middle initial	0.888	0.075
Year of birth	0.773	0.011
Month of birth	0.933	0.083
Day of birth	0.851	0.033
Province or country of birth	0.981	0.117

5.6 Conclusions

The simulations clearly demonstrate the equal importance of auxiliary variables based on the linking variables and high quality clerical reviews. Specifying good models is also important for the efficiency of the resulting estimators. However using the correct model is not required, because, like previous model-assisted estimators [53], the proposed estimators remain design-consistent even when the model is misspecified.

Table 5.2: Relative bias and CV for cell (0,0) for scenarios 1 through 3.

Scenario	Design	Estimator	Relative bias (%)	CV (%)
1	1	1	-0.12	7.52
		2	0.45	3.33
	2	1	0.34	1.52
		2	0.48	1.36
2	1	1	0.77	7.62
		2	0.94	6.43
	2	1	-0.17	5.67
		2	-0.29	5.44
3	1	1	0.18	25.18
		2	0.11	12.57
	2	1	0.32	6.79
		2	-0.04	6.37

Table 5.3: Relative bias and CV for cell (0,0) for scenarios 4 through 6.

Scenario	Design	Estimator	Relative bias (%)	CV (%)
4	1	1	1.21	7.71
		2	0.62	4.22
	2	1	0.25	2.40
		2	0.21	2.29
5	1	1	-4.94	8.25
		2	-5.25	3.66
	2	1	-6.31	14.84
		2	-6.23	14.79
6	1	1	-0.79	7.40
		2	-0.10	0.48
	2	1	-0.01	0.82
		2	0.01	0.12

Figure 5.1: Box plot of the relative bias for cell (0,0) in scenario 1. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.

Figure 5.2: Box plot of the relative bias for cell (0,0) in scenario 2. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.

Figure 5.3: Box plot of the relative bias for cell (0,0) in scenario 3. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.

Figure 5.4: Box plot of the relative bias for cell (0,0) in scenario 4. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.

Figure 5.5: Box plot of the relative bias for cell (0,0) in scenario 5. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.

Figure 5.6: Box plot of the relative bias for cell (0,0) in scenario 6. Estimator 1 is the HT estimator. Estimator 2 is the model-assisted estimator.

There are two potential issues with clerical reviews including the quality of the supporting information and the quality of the review process. Meaningful clerical reviews are obviously impossible unless the supporting information is sufficient and reliable. Even when it is the case, many questions remain about the quality of the review process and ways to objectively measure it. Indeed there are few studies on this subject, beyond that by Newcombe et al. [59]. Furthermore, such studies may be hard to replicate, either because they have not disclosed important methodological details, or because their results are heavily dependent on the used datasets that are unavailable. A second challenge is the development of anonymization techniques. They prevent clerical reviews and adversely impact the linking efficacy. Solutions based on privacy-preserving record linkage are being actively researched to address these problems [60]. However, in situations where clerical reviews have been effective (e.g., with available names, birthdates and addresses in the original files), it is still unclear whether these solutions offer competitive privacy-preserving alternatives to clerical reviews. A third challenge concerns missing values in the linked files. The problem arises because clerical reviews are expensive, such that it is desirable to avoid sampling pairs where some variables of interest are missing. Such missing variables represent an unusual form of item nonresponse, because it is known prior to sample selection.

Although the proposed estimators do not require an accurate mixture model, having a good estimator of the conditional match probability is still crucial. In this chapter we have shown how this information may be effectively used in the weighting stage or in the sampling design, when the covariates are categorical and low-dimensional. In the same setting, the accurate estimation of the conditional match probability with a clerical sample presents no major difficulty, even if this probability varies significantly¹

¹This may be an indication that the linkage is informative.

with covariates even after accounting for the comparison vector of a pair. Indeed, we can afford to estimate the condition match probability for each cross-classification of the analytical variables and the possible comparison vectors. However, the picture becomes different with continuous or high-dimensional covariates. Although the proposed weighting strategy still applies, we must now adapt both the estimation procedure for the conditional match probability and the sampling design. For the conditional match probability, a first strategy is to apply nonparametric methods that involve some smoothing and directly operate on all the analytical variables, e.g. local polynomial regression. A second strategy is to first apply a dimension reduction technique, e.g. Principal Components Analysis (PCA), to the covariates and use the few selected principal components as before, i.e. to define strata where the conditional match probability may be estimated accurately within the available resources. For the sampling design, we also have at least two options. The first option does not stratify the pairs but draw then with an inclusion probability that is proportional to their size according to the corresponding conditional match probability. Another option is to stratify the pairs based on the principal components for the covariates and to apply a Neyman allocation.

Chapter 6

Application

6.1 Introduction

In Canada, vital statistics registries and health surveys provide complementary data about public health. Vital statistics registries include the Canadian Mortality Database (CMDB) that provides mortality data by Cause of Death (CoD), demographic characteristics and province, but no information about important factors such as the lifestyle including smoking habits and physical activity levels. Although health surveys provide the latter information, they only do so on a cross-sectional basis and thus cannot provide any respondent health data beyond the survey reference date. This important limitation also applies to the Canadian Community Health Survey (CCHS), a national survey and the most important health survey in Canada, which was first conducted in 2000/2001. Sanmartin et al. [2] have addressed this issue by linking CCHS samples (from 2000 to 2011) to the CMDB (over the same reference period) and by fitting a Cox proportional hazards model (PHM) to the resulting data set, thus gaining some insight into factors of mortality, which are related to the lifestyle. A probabilistic linkage was used including an internal validation based on the evaluation of linkage errors through clerical-reviews. However, the analysis was

not adjusted for those errors. The results of the survival analysis are summarized in Table 6.1. They show higher hazard ratios (HRs) for mortality among groups that are at greater risk including heavy smokers and light smokers.

The study by Sanmartin et al. [2] provides the inspiration for our application. However, we depart from this work in many ways including the following two important assumptions:

1. The 2000/2001 CCHS sample is a simple random sample of some finite population.
2. This finite population generates each CMDB death record between 2000 and 2011.

Other differences concern the probabilistic linkage and the method of analysis that is applied to the resulting data set.

6.2 Data

For our study, the data sources include CMDB records from 2000 to 2011, and CCHS respondents for the 2000/2001 sample.

6.2.1 Canadian Mortality Database

The CMDB keeps a record of each death registered in Canada since 1950. For each death the information includes the death date, time of death, cause of death, names (including surnames and given names), birth date, and the postal code. This information is of high quality and available for the overwhelming majority of records¹.

¹Less than 1% missing for the last name, first given name, sex, birth date and death date, and less than 4.1% missing for the postal code, in the CMDB from 2000 to 2009

Table 6.1: Age-adjusted hazard ratios for mortality associated with selected health behaviours, by sex, respondents aged 20 or older to 2003 and 2005 Canadian Community Health Surveys linked to Canadian Mortality Database [2].

Health behaviour	Men			Women		
	Hazard ratio	95% CI		Hazard ratio	95% CI	
		from	to		from	to
Smoking						
Non-smoker	1	...	...	1	...	...
Light smoker	1.92*	1.51	2.33	1.81*	1.52	2.11
Heavy smoker	2.36*	1.84	2.89	2.91*	1.92	3.91
Former smoker	1.23*	1.05	1.4	1.31*	1.16	1.46
Body Mass Index (BMI), with correction						
Underweight (less than 18.5)	1.77*	1.06	2.47	1.5*	1.16	1.85
Normal weight (18.5 to less than 25.0)	1	...	...	1	...	...
Overweight (25.0 to less than 30.0)	0.87*	0.79	0.95	0.86*	0.78	0.94
Obese I (30.0 to less than 35.0)	0.96	0.85	1.07	0.91	0.8	1.02
Obese II (35.0 or more)	1.51*	1.25	1.76	1.2*	1	1.4
Alcohol consumption						
Light or non-drinker	1.2*	1.09	1.31	1.15*	1.01	1.29
Moderate drinker	1	...	...	1	...	...
Heavy drinker	1.35*	1.06	1.63	2.29	0.41	4.16
Former drinker	1.65*	1.46	1.83	1.56*	1.36	1.75
Minutes of physical activity per day						
None	1.89*	1.61	2.16	2.04*	1.65	2.43
Less than 30	1.23*	1.08	1.38	1.22*	1.03	1.41
30 to 60	0.99	0.86	1.12	1.1	0.92	1.28
More than 60	1	...	...	1	...	...
Fruit and vegetable servings per day						
None	1.93	0.49	3.36	1.71	0.57	2.85
Less than 2	1.52*	1.3	1.74	1.82*	1.51	2.13
2 to 5	1.18*	1.03	1.32	1.18*	1.05	1.32
More than 5	1	...	...	1	...	...

reference categories on blue lines

... not applicable

*significantly different from reference category ($p < 0.05$)

The postal code has a higher percentage of missing values than the other variables. To address this problem, Sanmartin et al. [2] have enriched their data, by conducting a preliminary linkage of the CMDB to historical tax files, to obtain additional postal code information. In our study, this step is not carried out.

To ease the computational burden, a Poisson sample of the CMDB is taken with a sampling fraction of 2%. A sampled record is kept if the variables sex, given name, surname, and birth date are nonmissing, and the death date is nonmissing and no earlier than January 1st 2001. Ultimately, 65,246 CMDB records are selected.

6.2.2 Canadian Community Health Survey

CCHS is a cross-sectional survey that collects data about health for Canadians aged 12 or older, who live in households and outside institutions (e.g., prisons, hospitals, etc.). It excludes full-time members of the Canadian Forces and residents of reserves and some remote areas². The 2000/2001 sample has a size of roughly 130,000. The content includes information on smoking habits, body mass index, alcohol consumption, physical activity, and diet (fruit and vegetables). Among the respondents³, 89.6% have given their consent to share their survey information with provincial and federal ministries of health and to link their responses to administrative data [2]. They form the CCHS records that are eligible for a linkage to the CMDB. For our study, we select the subset of these records where the variables sex, given name, surname, and birth date are nonmissing, and where the smoker type is not coded as "not stated". This results in the selection of 108,963 CCHS records.

The 2000/2001 CCHS survey was based on a sample of households, which were selected from three frames, including an area frame (83% of the sample), Random Digit

²Altogether 4% of the target population

³Between 2000 and 2011, CCHS response rates ranged from 69.8% to 78.9% [2]

Dialing (7% of the sample) and a list of telephone numbers (10% of the sample). The households were stratified by province or territory and by Health Region (HR), and the allocation was as in Table 6.2.

Table 6.2: Allocation for 2000/2001 CCHS sample.

Province/territory	Number of HRs	Total sample size (targeted)
Newfoundland	6	4,010
Prince Edward Island	2	2,000
Nova Scotia	6	5,040
New Brunswick	7	5,150
Quebec	16	24,280
Ontario	37	42,260
Manitoba	11	8,000
Saskatchewan	11	7,720
Alberta	17	14,200
British Columbia	20	18,090
Yukon	1	850
Northwest Territories	1	900
Nunavut	1	800
Canada	136	133,300

6.3 Probabilistic linkage

Sanmartin et al. [2] have implemented a probabilistic linkage with G-LINK⁴, where the linkage weights are set in a manual iterative manner. Then a given pair is linked if its weight exceeds a threshold. Following such decisions, conflicts may arise when the same CMDB record is linked to different CCHS records, or when two CMDB records

⁴Statistics Canada generalized system for probabilistic linkage

are linked to the same CCHS record. These conflicts are resolved in a *mapping* step, where decisions are made to undo some links, including some manual decisions.

In our case, a simpler probabilistic linkage methodology is implemented in SAS but outside G-LINK. As explained before, the goal is not to produce a linkage decision for each record-pair but to estimate its conditional match probability given its comparison vector. The estimated conditional match probability is computed under the assumption that the selected linkage variables are conditionally independent given the pair match status.

6.3.1 Variables

We use the variables last name, given name, sex, birth date⁵ as blocking or linkage variables⁶. However, in each input file, we only keep the subset of records that have no missing values in any of these variables.

6.3.2 Blocking criteria

Blocking criteria are required to obtain a manageable subset of record pairs, called *potential pairs*. We select the pairs where the last name, birth day⁷ and the sex are nonmissing and satisfy the following three conditions:

1. Same last name SOUNDEX code
2. Same birth day
3. Same sex

⁵The three components

⁶The postal code is not used because agreement on postal code is overemphasized at the expense of agreement on other variables. This is a problem in rural areas where postal codes cover large geographic areas

⁷Based on the day component in a birth date

Each combination of the last name and the birth day produces a distinct block. A total of 598,990 pairs⁸ are selected. In Sanmartin et al. [2], blocking is instead based on the phonetic⁹ agreement of (nonmissing) last names and the exact agreement of (nonmissing) birth dates.

6.3.3 Comparison vector

The comparison vector γ has four components, which are based on an exact agreement on the surname, given name, year of birth, and month of birth. Sanmartin et al. [2] use more elaborate comparisons that include partial agreements¹⁰.

6.3.4 Mixture model

The blocks are assumed perfect, i.e., each matched pair is assumed to fall within some block. Each pair is characterized by its comparison vector $\gamma = (\gamma^{(1)}, \dots, \gamma^{(K)}) \sim P(M)P(\gamma|M) + P(U)P(\gamma|U)$, where $\gamma^{(k)} = 1$ if there is a full agreement on the corresponding variable. Conditional independence is assumed. The overall mixing proportion $\alpha = P(M)$ is an unknown parameter in the interval $(0, 1)$. The parameter α and the conditional probabilities $[P(\gamma^{(k)} = 1|M)]_{1 \leq k \leq K}$ and $[P(\gamma^{(k)} = 1|U)]_{1 \leq k \leq K}$ are estimated by maximizing the marginal composite likelihood¹¹ over all the potential pairs. A quasi-Newton procedure is used that is implemented by calling a nonlinear optimization routine¹² in SAS IML. The conditional match probability of

⁸For men and women combined

⁹The NYSIIS code is used that assigns the same code to names that sound similar. It is tailored to European names.

¹⁰A partial agreement provides information about the nature of the differences when the values are not identical, e.g., a typo or a given similarity measured with Jaro-Winkler string comparison function.

¹¹This is not a traditional likelihood because the potential pairs are dependent.

¹²NLPNMS

a pair is estimated by

$$\hat{P}(M|\gamma) = \left(1 + \left(\frac{1}{\hat{\alpha}} - 1 \right) \frac{\hat{P}(\gamma|U)}{\hat{P}(\gamma|M)} \right)^{-1}.$$

6.4 Analysis

The analysis is based on the PW2 estimator that is detailed in the survival example of Section 4.5.2, under the assumption of an SRS design for the CCHS sample, as previously stated, i.e. $\pi(\cdot)$ is assumed constant. Recall that in this example, the survival probability is modeled using a proportional hazard model with an exponential baseline. The PW1 estimator is not used because the block sizes $N_1, \dots, N_H$ are not given. We consider the covariates smoker type (one of "daily", "occasional", "always occasional", "former daily", "former occasional", and "never smoked") and age on January 1, 2000. The smoking variable is recorded as a one "never smoked" (when the original answer is "never smoked") or "ever smoked" (in all other cases). A separate analysis is done for females and males. The analytical data set is the data set of potential pairs, where the estimated conditional match probability is no less than the threshold of 0.9. For men, a total of 81 pairs exceed this threshold. They agree on all four variables and have a conditional match probability of 0.996. For women, a total of 97 pairs meet this condition, including 87 pairs that agree on all variables with a conditional match probability of 0.993, and 10 pairs that agree on all variables except the month and birth (MM) with a conditional match probability of 0.943. The fitting is based on the numerical maximization of the composite likelihood in SAS IML using available nonlinear optimization routines. Finally, a bootstrap procedure is used to estimate the standard error of each parameter. It operates by resampling the blocks (assumed to be IID). Twenty bootstrap samples are generated. For each sample,

the linkage parameters and survival parameters are recomputed. The results are described in the next section.

6.5 Results

The mixture parameters are estimated with a SAS IML quasi-Newton procedure that converges in 67 iterations and 47 iterations for men and women, respectively. Table 6.3 shows the estimates that include the mixing proportions (for men and women) and the agreement probabilities (matched and unmatched) for each variable. The results are similar for men and women. The estimated mixing-proportions are small (0.053% and 0.047% for men and women, respectively) and well below the 5% threshold that is believed to be required for the convergence of the E-M procedure. The corresponding standard errors are large relative to the point estimates. This is not surprising because the actual mixing-proportions are expected to be small proportions. The agreement probabilities are much larger for matched pairs than for unmatched pairs¹³. Indeed, for the given name, the birth year (YY) and the birth month (MM), the agreement probability is roughly two orders of magnitude larger than in an unmatched pair. For the surname, the odds ratio is much smaller because the surname is used in the blocking criteria. Consequently, an unmatched pair has a much higher chance of agreeing on the surname than a random pair in the Cartesian product of the two files. For men, all agreement probabilities are close to 1 for a matched pair. For women, the agreement probability is close to 1 for the surname and the given name, but much smaller for the birth date components (about 0.62 and 0.44 for the year and month of birth, respectively). Overall, these results indicate that the selected linkage variables provide a good discrimination between the matched and the unmatched pairs.

¹³These are the unmatched pairs in the blocks.

Table 6.3: Estimated mixture parameters.

Parameter	Variable	Men		Women	
		Estimate	SE	Estimate	SE
α	...	0.00053	0.00052	0.00047	0.0013
$P(\gamma^{(k)} = 1 M)$	Surname	0.9667	0.02489	0.9835	0.0438
	Given name	0.8366	0.20268	0.9514	0.2686
	YY	0.9851	0.34176	0.6196	0.26
	MM	0.9356	0.27533	0.4443	0.2588
$P(\gamma^{(k)} = 1 U)$	Surname	0.2995	0.01289	0.3124	0.0126
	Given name	0.0068	0.00052	0.0036	0.0002
	YY	0.0096	0.00024	0.0087	0.0003
	MM	0.0818	0.00085	0.0836	0.0006

...not applicable

The estimated regression coefficients are given in Table 6.4. For men, the estimated coefficient for smoking is positive but the 95% confidence interval is quite large and does include 0. For women, the corresponding estimate is negative, with a large 95% confidence interval that also includes 0. Thus we cannot conclude that having smoked has an impact on the hazard ratio, which is counterintuitive and inconsistent with the previous study by Sanmartin et al. [2]. There are many possible reasons including the omission of important covariates (e.g., alcohol consumption, body mass index, or physical activity level), the small sample size due to the 2% sampling of the CMDDB (with only 81 and 97 pairs above the 0.9 threshold for men and women respectively), and our simplifying assumption about the CCHS sample design.

Table 6.4: Estimated regression coefficients.

Coefficient	Men				Women			
	Estimate	SE	95% CI		Estimate	SE	95% CI	
			from	to			from	to
Intercept	-12.46	1.34	-15.09	-9.83	-7.96	3.23	-14.29	-1.63
Age	-0.2	0.06	-0.32	-0.08	0.06	0.08	-0.1	0.22
Ever smoked	2.41	1.83	-1.18	6	-3	5.77	-14.31	8.31

For smoking, "Never smoked" is used as reference category.

Chapter 7

Conclusions

The accurate analysis of linked data is an important problem in official statistics. In this work, we have described two general methodologies for doing so when the analytical data set is based on the linkage of two data sources about the same finite population, and while explicitly accounting for linkage errors, or more accurately the uncertainty about the match status of record pairs. Both methodologies require a mixture model for the marginal distribution of agreements that are observed in a record pair. The first methodology is model-based and requires no clerical-review. However, it relies on different assumptions of conditional independence (given the covariates) among the responses, the linkage variables, and the different selection mechanisms if the sources are samples instead of registers. The resulting estimators are biased if the marginal pair mixture model is misspecified. The second methodology is model-assisted (see Sarndal et al. [53]) and depends on the availability of a clerical-review sample. It produces consistent estimators even if the marginal pair mixture model is misspecified. These two solutions complement each other and may be considered according to the available resources. Although these methods give encouraging simulation results, both of them can be improved in various ways that represent interesting directions for future research.

For the model-assisted-EEs, the central question is the reliability of the clerical reviews, which may be addressed by combining repeated reviews and latent class analysis [21].

The model-based EEs rely on many assumptions regarding the relationship between the responses, the comparison vectors and the selection mechanisms in the two files. However no clerical sample is required. This is a major advantage because procuring this sample may be an expensive and difficult task. Two kinds of model-based EEs are described. The first kind is based on the expression of the marginal expectation of recorded responses given all the covariates in the corresponding block, and the comparison vector of an *adjacent* pair, i.e. a pair involving the corresponding record. The second kind is based on the expression of the marginal expectation of recorded responses given the covariates from a record (in the same block) in the other file, and the comparison vector for the corresponding pair. Both kinds of EEs lead to consistent estimators that may be computed by weighted least squares or the maximization of a composite likelihood; the latter case occurring when the original responses have a conditional distribution with a known parametric form. The first EEs lead to more efficient estimators but they are less convenient in any setting when two samples are linked, because there is a need to know the population size of each block. In such cases, the second EEs may be preferred even if the loss of efficiency can be quite large. Regarding the model-based EEs, future research may look at more comprehensive evaluation studies, the testing of underlying assumptions, the choice of optimal weights when using weighted least-squares, the properties of the estimators under realistic blocking criteria, and extensions for multi-linkages.

The future development of tests and diagnosis for the underlying assumptions is crucial.

The problem of optimal weight selection of weight is another important problem that

arises when using weighted least squares. The challenge is the following crucial difference between the pairwise EEs and the traditional EEs of quasi-likelihood theory [47]. In this theory, the EEs are traditionally based on a weighted sum of mean responses (or conditional mean responses given *random* covariates) across independent clusters, with correlated responses in each cluster. In this case, the best estimator (i.e. the one with the smallest asymptotic variance) is produced by choosing the weights according to the variance-covariance¹ of the responses, in each cluster. However, the proposed pairwise EEs involve random block covariates and a weighted sum of conditional mean responses, with a different conditioning event for each conditional mean in a cluster. Indeed, in Theorem where we consider two linked registers, block h contributions involve the conditional means $[E[z_j | N_h, [\mathbf{x}_{i'}]_{i' \in A_h}, \gamma_{ij}]]_{1 \leq i, j \leq N_h}$, and the conditioning events $[\{N_h, [\mathbf{x}_{i'}]_{i' \in A_h}, \gamma_{ij}\}]_{1 \leq i, j \leq N_h}$ respectively. In this case, the weight related to the conditional mean $E[z_j | N_h, [\mathbf{x}_{i'}]_{i' \in A_h}, \gamma_{ij}]$ must be a function of $N_h, [\mathbf{x}_{i'}]_{i' \in A_h}$, and γ_{ij} , lest the EE becomes biased. Under this constraint, the choice of optimal weights is a greater challenge than in the quasi-likelihood theory. In previous chapters, the proposed solutions choose the optimal weights that would be assigned if the different contributions were independent within each block, i.e. an independent working correlation matrix is used. This is a good choice if the linkage variables provide enough information and within each block, we only select the pairs that have a sufficiently high conditional match probability or expected conditional match probability, i.e. setting $\tau_{ij} = I(q_{ij} \geq q_0)$ (where q_0 is close to 1, e.g. $q_0 = 0.9$) when using all block covariates and setting $\tau_{ij} = I(E[q_{ij} | O_{ij}] \geq q_0)$ when using only using the covariates in a pair. Indeed, within each block, each selected pair has a high probability of being matched, and matched pairs are mutually independent, if the original population is comprised of IID individuals, the two files are free of duplicates, and each individual is included

¹i.e. the conditional variance-covariance given the covariates, if the covariates are random.

in the two files independently of the other individuals, as we have assumed. Yet finding the optimal weights would be useful in situations where the linkage variables do not provide enough information and the conditional match probability q_{ij} is bounded away from 1 by a wide margin, e.g. if $q_{ij} \leq 0.7$ for any possible comparison vector². Another challenge is the consideration of realistic blocking criteria. In this thesis, the blocks have been assumed IID and perfect³, with block sizes that are bounded above by a constant regardless of the number of blocks H or the population size $N = O(H)$. With such criteria, the subset of selected pairs is the union $\bigcup_{h=1}^H A_h \times B_h$ of smaller Cartesian products, where $A_h \times B_h$ is the Cartesian product within block h , while $[A_h]_{1 \leq h \leq H}$ and $[B_h]_{1 \leq h \leq H}$ are two partitions of $\{1, \dots, N\}$. However, realistic blocking criteria are usually imperfect, i.e. some matched pairs may not satisfy them. Also, the subset of selected pairs can be any subset of the Cartesian product $A \times B$. The model-based EEs may be extended for such general criteria. As an illustration, consider a finite population with N individuals and two (duplicate-free) registers, where the available linkage information⁴ grows with N ⁵. As before, let γ_{ij} denote the comparison vector for the pair $(i, j) \in \{1, \dots, N\}^2$, that now belongs to the set Γ_N of all possible comparison vectors. Suppose that the blocking criteria are now based on the condition $\gamma_{ij} \in \mathcal{P}_N$ for some subset $\mathcal{P}_N$ of Γ_N ⁶, such that $P(\gamma_{ij} \in \mathcal{P}_N | m_{ij} = 1) = O(1)$ and $P(\gamma_{ij} \in \mathcal{P}_N | m_{ij} = 0) = O(N^{-1})$ as $N \rightarrow \infty$. Thus the expected number of selected pairs is $NP(\gamma_{ij} \in \mathcal{P}_N | m_{ij} = 1) + N(N-1)P(\gamma_{ij} \in \mathcal{P}_N | m_{ij} = 0) = O(N)$ instead of

²In such a case, a relevant question is whether one should proceed with the analysis of the linked data set or look for other avenues to produce the needed statistical information.

³The blocks are perfect if matching records always belong to the same block.

⁴In practice, the information of a (categorical) linkage variable is measured by its Shannon entropy $E[-\log p(v)] = -\sum_v p(v) \log p(v)$, where $p(v)$ is the probability of observing the value v . The information provided by the variable is large if its entropy is large.

⁵For example, imagine a population of IID individuals where each individual is characterized by $O(\log_2 N)$ IID linkage variables, and where the recording errors on these variables are conditionally independent given the original variables.

⁶For example, if we have $\lceil \log_2 N \rceil$ linkage variables and define the comparison vector based on perfect agreement for each variable, Γ_N comprises of all the $\lceil \log_2 N \rceil$ -dimensional binary vectors. Then $|\Gamma_N| = 2^{\lceil \log_2 N \rceil}$.

the N^2 pairs of the Cartesian product. With minor changes, we can apply Theorem in the special case where $H = 1$, $N_1 = N$, $A_1 = A$ and $B_1 = B$, if the assumptions of Section (3.3) still hold and the distributions of the covariates and responses do not depend on the population size. Then for each selected pair (i, j) (i.e. where $\gamma_{ij} \in \mathcal{P}_N$), Eq. (3.1) still applies. For the estimation procedures of Section 3.5, we can set τ_{ij} to zero outside $\mathcal{P}_N$. Within $\mathcal{P}_N$, we may set $\tau_{ij} = I(q_{ij} \geq q_0)$ as before. Studying the large sample behaviour of the resulting estimators is a greater challenge because the resulting EEs are no longer based on IID sums but involve U-statistics, where the kernels depend on N . To address this challenge, we can look into the theory of U-statistics.

The pairwise EEs may be extended for linkage environments that are used to expedite the production of analytical datasets through multiple consecutive linkages, where three or more files are linked. A good example is the Social Data Linkage Environment (SDLE) at Statistics Canada. The SDLE is based on a backbone or spine to which various social data sets (survey or administrative data) are linked in sequence, using names, demographic variables, postal codes,... The resulting linkage keys identify the records that are deemed matched across all the sources that have been linked to the backbone. For confidentiality reasons, the linkage keys and analytical variables are stored separately. With the SDLE, the linkage effort scales linearly with the number of sources that must be linked to produce a given analytical data set. For multiple linkages with three or more files, complex scenarios may arise according to the number of sources, and the distribution of the covariates and the responses across the different sources. However, when looking at extending the model-based EEs, we may first consider the following simple scenario that is motivated by the SDLE setting. Suppose that the analytical data set must be produced by linking

a number of *satellite* registers to a *backbone* register, for the same population. One satellite register contains the responses, while the covariates are partitioned among the other satellite registers and the backbone register. Thus each covariate is found in exactly one register⁷. As before, consider an IID population where individual i is characterized by a vector of linkage variables, responses $\mathbf{y}_i$ and a vector of covariates $\mathbf{x}_i$. Rewrite this vector rewritten as $\left[\mathbf{x}_i^{(0)} \mathbf{x}_i^{(1)} \dots \mathbf{x}_i^{(R-1)}\right]$, where R is the number of satellite registers, $\left[\mathbf{x}_i^{(0)} \mathbf{x}_i^{(1)}\right]$ are the covariates in the backbone register, and $\mathbf{x}_i^{(r+1)}$ are the covariates in the satellite register r where $r = 1, \dots, R-1$. Satellite register R contains the recorded responses $\mathbf{y}_i$ that is stored as $\mathbf{z}_j$ for some $j = 1, \dots, N$. Suppose that the same blocks apply when linking each satellite register to the backbone. These blocks are described in terms of the partitions $\left[A_h^{(0)}\right]_{1 \leq h \leq H}, \dots, \left[A_h^{(R-1)}\right]_{1 \leq h \leq H}$, and $[B_h]_{1 \leq h \leq H}$ of $\{1, \dots, N\}$, such that the pairs $\bigcup_{h=1}^H A_h^{(0)} \times A_h^{(r)}$ are selected by the blocks for the linkage of satellite r ($r = 1, \dots, R-1$) to the backbone, and the pairs $\bigcup_{h=1}^H A_h^{(0)} \times B_h$ are selected by the blocks for the linkage of satellite R (with the recorded responses $\mathbf{z}_j$) to the backbone. For $r = 1, \dots, R$, let $m_{(ij)}^{(r)}$ and $\gamma_{ij}^{(r)}$ denote the match status and comparison vector between record i from the backbone and record j from the satellite r , respectively. In block h , define the match matrix for the linkage of satellite r by $\mathbf{M}_h^{(r)} = \left[m_{(ij)}^{(r)}\right]_{(i,j) \in A_h^{(0)} \times A_h^{(r)}}$ if $r = 1, \dots, R-1$, and by $\mathbf{M}_h^{(R)} = \left[m_{(ij)}^{(R)}\right]_{(i,j) \in A_h^{(0)} \times B_h^{(R)}}$. Now suppose the conditional independence of $\left[\mathbf{x}_i^{(1)}\right]_{i \in A_h^{(0)}}$, $\mathbf{M}_h^{(1)}$, $\left[\mathbf{x}_i^{(2)}\right]_{i \in A_h^{(1)}}$, $\dots$, $\mathbf{M}_h^{(R-1)}$, $\left[\mathbf{x}_i^{(R)}\right]_{i \in A_h^{(R-1)}}$, $\mathbf{M}_h^{(R)}$, and $[\mathbf{y}_i]_{i \in B_h}$, given $\left[\mathbf{x}_i^{(0)}\right]_{i \in A_h^{(0)}}$, in block h for $h = 1, \dots, H$. In this setting, We can try and work out the details of this extension by leveraging the stated assumptions and previous work on record-groups [14–16].

⁷Note that this assumption is not restrictive, since we can always choose the source register for a covariate that appears on many registers.

Finally more comprehensive simulation studies may be conducted including comparisons with joint models and Bayesian solutions for analysing linked data [12, 13].

List of References

- [1] H. Newcombe, *Handbook of Record Linkage*. New York: Oxford University Press, 1988.
- [2] C. Sanmartin, Y. Decady, R. Trudeau, A. Dasylva, M. Tjepkema, P. Finés, R. Burnett, N. Ross, and D. Manuel, “Linking the canadian community health survey and the canadian mortality database: An enhanced data source for the study of mortality,” in *Health Reports*, vol. 27 of *Catalogue no. 82-003-X*, pp. 1–11, Statistics Canada, 2016.
- [3] T. Herzog, F. Scheuren, and W. Winkler, *Data Quality and Record Linkage Techniques*. New York: Springer, 2007.
- [4] T. Herzog, F. Scheuren, and W. Winkler, *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution and Duplicate Detection*. Berlin: Springer, 2012.
- [5] I. Fellegi and A. Sunter, “A theory of record linkage,” *Journal of the American Statistical Association*, vol. 64, pp. 1183–1210, 1969.
- [6] D. Da Silveira and E. Artmann, “Accuracy of probabilistic record linkage applied to health databases: systematic review,” *Rev Saude Publica*, vol. 43, 2009.
- [7] M. Bohensky, D. Jolley, V. Sundararajan, S. Evans, D. Pilcher, I. Scott, and C. Brand, “A powerful research tool with potential problems,” *BMC Health Services Research*, vol. 10, pp. 1–7, 2010.
- [8] Statistics Canada, ed., *Record Linkage Project Process Model*. **Catalog no 12-605-X**, Statistics Canada, 2017.
- [9] K. Wilkins, M. Shields, and M. Rotermann, “Smokers’ use of acute care hospitals a prospective study,” in *Health Reports*, vol. 20 of *Catalogue no. 82-003-X*, pp. 1–9, Statistics Canada, 2009.

- [10] Australian Bureau of Statistics, “Australians’ journeys through life: Stories from the australian census longitudinal dataset, 2006 - 2011,” in *Information Paper, catalogue no. 2081.0*, Australian Bureau of Statistics, 2013.
- [11] J. Neter, E. Maynes, and R. Ramanathan, “The effect of mismatching on the measurement of response errors,” *Journal of the American Statistical Association*, vol. 60, pp. 1005–1027, 1965.
- [12] M. Fortini, B. Liseo, A. Nuccitelli, and M. Scanu, “On bayesian record linkage,” *Research in Official Statistics*, vol. 4, pp. 185–198, 2001.
- [13] A. Tancredi and B. Liseo, “A hierarchical bayesian approach to record linkage and population size problems,” *Annals of Applied Statistics*, vol. 5, pp. 1553–1585, 2011.
- [14] M. Sadinle and S. Fienberg, “A generalized fellegi-sunter framework for multiple record linkage with application to homicide record systems,” *Journal of the American Statistical Association*, vol. 108, pp. 385–397, 2013.
- [15] M. Sadinle, “Detecting duplicates in a homicide registry using a bayesian partitioning approach,” *Annals of Applied Statistics*, vol. 8, pp. 2404–2434, 2014.
- [16] M. Sadinle, “Bayesian estimation of bipartite matchings for record linkage,” *Journal of the American Statistical Association*, vol. 112, pp. 600–612, 2017.
- [17] P. McCullagh and J. Nelder, *Generalized Linear Models*. New York: Chapman and Hall, 1983.
- [18] A. Agresti, *Categorical Data Analysis*. Hoboken: Wiley, 2002.
- [19] M. Jaro, “Advances in record linkage methodology to matching the 1985 census of tampa, florida,” *JASA*, vol. 84, pp. 414–420, 1989.
- [20] A. Dasylva, “Design-based estimation with record-linked administrative files,” in *Proceedings of the 2014 International Methodology Symposium*, 2014.
- [21] A. Dasylva, M. Abeysondera, B. Akpoue, M. Haddou, and A. Saidi, “easuring the quality of a probabilistic linkage through clerical reviews,” in *Proceedings of the 2016 International Methodology Symposium*, 2016.
- [22] R. Chambers, “Regression analysis of probability-linked data,” in *Research Series in Official Statistics*, Government of New Zealand, 2009.

- [23] Y. Thibaudeau, “The discrimination power of dependency structures in record linkage,” *Survey Methodology*, vol. 19, pp. 31–38, 1993.
- [24] F. Scheuren and W. Winkler, “Regression analysis of data that are computer matched,” *Survey Methodology*, vol. 19, pp. 39–58, 1993.
- [25] F. Scheuren and W. Winkler, “Regression analysis of data that are computer matched - part ii,” *Survey Methodology*, vol. 23, pp. 157–165, 1997.
- [26] P. Lahiri and D. Larsen, “Regression analysis with linked data,” *Journal of the American Statistical Association*, vol. 100, pp. 222–227, 2005.
- [27] J. Chipperfield, G. Bishop, and P. Campbell, “Maximum likelihood estimation for contingency tables and logistic regression with incorrectly linked data,” *Survey Methodology*, vol. 37, pp. 13–24, 2011.
- [28] J. Chipperfield and R. Chambers, “Using the bootstrap to analyse binary data obtained via probabilistic linkage,” *Journal of Official Statistics*, vol. 31, pp. 397–414, 2015.
- [29] D. Krewski, A. Dewanji, Y. Wang, S. Bartlett, J. Zielinski, and R. Mallick, “The effect of record linkage errors on risk estimates in cohort mortality studies,” *Survey Methodology*, vol. 31, pp. 13–22, 2001.
- [30] R. Mallick, *Assessment of record-linkage and measurement error in cohort mortality studies*. PhD thesis, Carleton University, 2005.
- [31] M. Hof, A. Ravelli, and A. Zwinderman, “A probabilistic linkage model for survival data,” *Journal of the American Statistical Association*, vol. 112, pp. 1504–1515, 2017.
- [32] J. Wang and P. Donnan, “Adjusting for missing record-linkage in outcome studies,” *Journal of Applied Statistics*, vol. 29, pp. 873–884, 2002.
- [33] Y. Ding and S. Fienberg, “Dual system estimation of census undercount in the presence of matching error,” *Survey Methodology*, vol. 20, pp. 149–158, 1994.
- [34] L. Di Consiglio and T. Tuoto, “Coverage evaluation on probabilistically linked data,” *Journal of Official Statistics*, vol. 31, pp. 415–429, 2015.
- [35] R. Chambers, J. Chipperfield, W. Davis, and M. Kovacevic, “Inference based on estimating equations and probability-linked data,” in *Research Series in Official Statistics*, University of Wollongong, 2009.

- [36] G. Kim and R. Chambers, “Regression analysis under incomplete linkage,” *Computational Statistics and Data Analysis*, vol. 56, pp. 2756–2770, 2012.
- [37] G. Kim and R. Chambers, “Regression analysis under probabilistic multi-linkage,” *Statistica Neerlandica*, vol. 66, pp. 64–79, 2012.
- [38] G. Kim and R. Chambers, “Bias reduction for correlated linkage error,” in *NI-ASRA Working Papers Series*, University of Wollongong, 2013.
- [39] P. Lahiri and J. Law, “Analysis of statistical models with linked data,” in *4th Baltic-Nordic Conference on Survey Statistics (BANOCOSS2015)*, 2015.
- [40] G. Kim and R. Chambers, “Secondary analysis of linked data,” in *Methodological developments in data linkage* (K. Harron, H. Goldstein, and C. Dibben, eds.), pp. 83–108, Chichester:Wiley, 2016.
- [41] M. Larsen, “Multiple imputation analysis of records linked using mixture models,” in *SSC Annual Meeting, Proceedings of the Survey Methods Section*, pp. 65–71, 2015.
- [42] H. Goldstein, H. Harron, and A. Wade, “The analysis of record-linked data using multiple imputation with data value priors,” *Statistics in Medicine*, vol. 21, pp. 1485–1496, 2015.
- [43] M. Hof and A. Zwiderman, “A mixture model for the analysis of data derived from record linkage,” *Statistics in Medicine*, vol. 34, pp. 74–92, 2012.
- [44] M. Hof and A. Zwiderman, “Methods for analyzing data from probabilistic linkage strategies based on partially identifying variables,” *Statistics in Medicine*, vol. 31, pp. 4231–4242, 2012.
- [45] A. Tancredi and B. Liseo, “Regression analysis with linked data: problems and solutions,” *Statistica*, vol. 75, 2015.
- [46] P. Billingsley, *Probability and Measure*. New York: Wiley, 1995.
- [47] C. Heyde, *Quasi-likelihood and its applications*. New York: Springer, 1997.
- [48] C. Varin, N. Reid, and D. Firth, “An overview of composite likelihood methods,” *Statistica Sinica*, vol. 21, pp. 5–42, 2011.
- [49] A. Van der Vaart, *Asymptotic Statistics*. Cambridge: Cambridge University Press, 1998.

- [50] A. Dasylva, “Design-based estimation with record-linked administrative files and a clerical review sample,” *Journal of Official Statistics*, vol. 34, pp. 41–54, 2018.
- [51] A. Dasylva, R.-C. Titus, and C. Thibault, “Overcoverage in the 2011 canadian census,” in *Proceedings of the 2014 International Methodology Symposium*, 2014.
- [52] Statistics Canada, ed., *2011 Census Technical Report series: Coverage*. Catalog no 98-303-X2011001, Statistics Canada, 2015.
- [53] C.-E. Sarndal, B. Swensson, and B. Wretman, *Model Assisted Survey Sampling*. New York: Springer, 1992.
- [54] J.-C. Deville and C.-E. Sarndal, “Calibration estimators in survey sampling,” *Journal of the Royal Statistical Society Series B*, vol. 37, pp. 376–382, 1992.
- [55] P. Lavalée, *Le Sondage indirect ou la mthode du partage des poids*. Bruxelles: Éditions de l’Université de Bruxelles, 2002.
- [56] W. Winkler, “Using the em algorithm for weight computation in the fellegi-sunter model of record linkage,” in *Proceedings of the Section on Survey Research Methods, ASA*, pp. 65–71, 1988.
- [57] A. Dempster, N. Laird, and D. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the Royal Statistical Society Series B*, vol. 39, pp. 1–38, 1977.
- [58] J. Jiang, *Linear and Generalized Linear Mixed Models and Their Applications*. New York: Springer, 2007.
- [59] H. Newcombe, M. Smith, and G. Howe, “Reliability of computerized versus manual death searches in a study of the health of eldorado uranium workers,” *Computers in Biology and Medecine*, vol. 13, pp. 157–169, 1983.
- [60] R. Schnell, T. Bachteler, and J. Reiher, “Privacy-preserving record linkage using bloom filters,” *BioMed Central Medical Informatics and Decision Making*, vol. 9, 2009.
- [61] J. Heyde, *Matrix Analysis for Statistics*. New York: Wiley, 1997.

Appendix A

Mathematical background

A.1 Stochastic orders of magnitude

Consider a random sequence $[X_n]_n$ and a deterministic sequence $[a_n]_n$. Write $X_n = O_p(a_n)$, if, for any $\epsilon > 0$, there exists a finite positive M and a finite positive integer N such that we have the upperbound

$$P(|X_n/a_n| > M) < \epsilon, \quad \forall n > N.$$

Write $X_n = o_p(a_n)$, if, for any $\epsilon > 0$, we have the limit

$$\lim_{n \rightarrow \infty} P(|X_n/a_n| \geq \epsilon) = 0.$$

Now consider two random sequences $[X_n]_n$ and $[Y_n]_n$. Write $Y_n = o_p(X_n)$ if $Y_n = R_n X_n$, where $R_n \xrightarrow{p} 0$. Write $Y_n = O_p(X_n)$ if $Y_n = R_n X_n$, where $R_n = O_p(1)$.

A.2 Matrix derivatives

We define matrix derivatives according to Schott [61, chap. 22]. Let $\boldsymbol{\theta} \in R^p$ and $\mathbf{f}(\boldsymbol{\theta}) = [f_1(\boldsymbol{\theta}) \dots f_r(\boldsymbol{\theta})]^\top$. Then define

$$\frac{\partial \mathbf{f}}{\boldsymbol{\theta}^\top} = \begin{bmatrix} \frac{\partial f_1}{\partial \theta_1} & \cdots & \frac{\partial f_1}{\partial \theta_p} \\ \vdots & & \\ \frac{\partial f_r}{\partial \theta_1} & \cdots & \frac{\partial f_r}{\partial \theta_p} \end{bmatrix} \quad (\text{A.1})$$

Appendix B

Code

B.1 Chapter 3

B.1.1 Linear regression

The following R code was used.

```
#-----#
#
# randomPermutation(blockSize=)
#
#-----#

randomPermutation = function (blockSize)
{
    u = runif(blockSize,0,1);
    sortedU = sort(u);

    #i=perm(j)

    permutationMatrix = matrix(rep(0,blockSize^2),blockSize ,blockSize);

    for (j in 1:blockSize)

        for (i in 1:blockSize) permutationMatrix[i,j] = (u[i]==sortedU[j]);

    return(permutationMatrix);
}

#-----#
#
```



```

# generateFinitePopulation (numBlocks=, blockSize=, numLinkVars=)
#
#-----#

generateFinitePopulation = function (numBlocks,
                                     blockSize,
                                     numLinkVars) # all binary variables
{

  ALPHA = 0.5;
  BETA = 1.0;
  MEAN_X = 0.0;
  SIGMA_X = 1.0;
  #num_x_steps = 10;
  SIGMA = 0.7;
  P = 0.5;
  Q0 = 0.05;
  Q1 = 0.95;
  SHUFFLEPROBA = 1.0;

  #for low quality Q0=0.2, Q1=0.8
  #for medium quality Q0=0.1, Q1=0.9
  #for high quality Q0=0.05, Q1=0.95

  popSize = numBlocks*blockSize;

  blocks = c(t(matrix(rep(c(1:numBlocks),blockSize),numBlocks,blockSize)));

  mixingProportion = 1/blockSize;

  uAgree = (P*Q1+(1-P)*Q0)^2+(1-(P*Q1+(1-P)*Q0))^2;
  mAgree = P*(Q1^2+(1-Q1)^2)+(1-P)*((1-Q0)^2+Q0^2);

  #uAgree = 1/4;
  #mAgree = 1/2;

  x = rnorm(popSize,MEAN_X, SIGMA_X);
  #x = round(-(num_x_steps/2)+num_x_steps*runif(popSize,0,1),0)/(num_x_steps/2);
  y = ALPHA+BETA*x+SIGMA*rnorm(popSize,0,1);

  origLinkVars = matrix(rbinom(popSize*numLinkVars,1,P),popSize,numLinkVars);

  recordedLinkVarsA = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars,1,Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars,1,Q1),popSize,numLinkVars);
  recordedLinkVarsB = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars,1,Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars,1,Q1),popSize,numLinkVars);

  blockids = c(t(matrix(rep(c(1:numBlocks),blockSize), numBlocks, blockSize)));

  recidA = cbind(blockids, matrix(rep(matrix(c(1:blockSize), blockSize, 1),numBlocks),popSize,1));
  #oRecidB = recidA;

```

```

recidB = recidA;

#-----#
# Apply a random permutation to B records
#-----#

# Shuffle the linkage variables and the responses
# within each block

#recidB = oRecidB;
oRecidB = recidB;

matchMatrices = list();

for(b in 1:numBlocks) {

  startIndex = (b-1)*blockSize+1;
  endIndex = b*blockSize;
  permMat = diag(rep(1,blockSize));

  shuffleYes = rbinom(1,1,SHUFFLEPROBA);

  if(shuffleYes) {
    permMat = randomPermutation(blockSize);
    oRecidB[startIndex:endIndex,1:2] = permMat%%recidB[startIndex:endIndex,1:2];
    recordedLinkVarsB[startIndex:endIndex,1:numLinkVars] = permMat%%recordedLinkVarsB[startIndex:endIndex,1:numLinkVars];
    y[startIndex:endIndex] = permMat%%y[startIndex:endIndex];
  }

  matchMatrices[[b]] = permMat;
}

FPData = list(
  numBlocks = numBlocks,
  blockSize = blockSize,
  popSize = popSize,
  numLinkVars = numLinkVars,
  blocks = blocks,
  recidA = recidA,
  oRecidB = oRecidB,
  recidB = recidB,
  matchMatrices = matchMatrices,
  origLinkVars = origLinkVars,
  recordedLinkVarsA = recordedLinkVarsA,
  recordedLinkVarsB = recordedLinkVarsB,
  shuffleProba = SHUFFLEPROBA,
  p = P,
  q0 = Q0,
  q1 = Q1,

```

```

        mixingProportion = mixingProportion,
        mAgree = mAgree,
        uAgree = uAgree,
        x = x,
        y = y,
        model = 'linear regression',
        alpha = ALPHA,
        beta = BETA,
        sigma = SIGMA,
        meanX = MEAN_X,
        sigmaX = SIGMA_X);

    return(FPData);
}

#-----#
#
# compareLinkVars(v1, v2)
#
#-----#

compareLinkVars = function(v1, v2) {

    numLinkVars = dim(v1)[2];
    numPairs = dim(v1)[1]

    gammas = matrix(rep(0,numLinkVars*numPairs),numPairs,numLinkVars);

    for(i in 1:numPairs)
        for(j in 1:numLinkVars) gammas[i,j] = (v1[i,j]==v2[i,j]);

    return(gammas);
}

#-----#
#
# generatePairs(FPData)
#
#-----#

generatePairs = function(FPData)
{

    popSize      = FPData$popSize;
    numBlocks    = FPData$numBlocks;
    blockSize    = FPData$blockSize;
    numLinkVars  = FPData$numLinkVars;
    recidA       = FPData$recidA;
    recidB       = FPData$recidB;
    oRecidB      = FPData$oRecidB;
    target_fnr   = 0.05;

```

```

mixingProportion = FPData$mixingProportion;

mAgree = FPData$mAgree;
uAgree = FPData$uAgree;

shuffleProba = FPData$shuffleProba;

ncols = 2+numLinkVars;

#-----#
# tableA: with ncols columns
#       col 1: (perfect) blocking variable
#       col 2 through col ncols-1: linkage variables
#       col ncols: covariate x
#
# tableB: with ncols columns
#       col 1: (perfect) blocking variable
#       col 2 through col ncols-1: linkage variables
#       col ncols: covariate y
#-----#

tableA = cbind(FPData$blocks, FPData$recordedLinkVarsA, FPData$x);
tableB = cbind(FPData$blocks, FPData$recordedLinkVarsB, FPData$y);

for(b in 1:numBlocks) {

  # Get the records in the block b in each file

  blockA=matrix(tableA[(tableA[,1]==b),], blockSize, ncols);
  blockB=matrix(tableB[(tableB[,1]==b),], blockSize, ncols);

  #if(0){
  startIndex_b=(b-1)*blockSize+1;
  endIndex_b=b*blockSize;

  oRecidB_b=matrix(rep(0,blockSize),blockSize,1);
  oRecidB_b=oRecidB[startIndex_b:endIndex_b,2];
  #}

  #print(list(oRecidB_b=oRecidB_b));

  for(r in 1:blockSize) {

    #-----#
    # For t the block size build
    #
    #       x_r y_1
    #       x_r y_2
    #       .   .
    #       .   .
    #       .   .
    #       x_r y_t
    #-----#

```

```

analyticalVars = cbind(matrix(rep(blockA[r,ncols],blockSize),blockSize,1), blockB[,ncols]);

#-----#
# For t the block size build
# v_j the vector of linkage
#   variables for record j
#
#   gamma(v_r,v_1)
#   gamma(v_r,v_2)
#   .   .
#   .   .
#   .   .
#   gamma(v_r,v_t)
#-----#

gammas = compareLinkVars(t(matrix(rep(blockA[r,2:(ncols-1)],blockSize),numLinkVars, blockSize)
), blockB[,2:(ncols-1)]);
tmpMat = cbind(rep(b,blockSize), rep(r,blockSize), c(1:blockSize), gammas, analyticalVars,
matrix(rep(0,5*blockSize),blockSize,5), 1*(oRecidB_b==r));

if(b==1 && r==1) potentialPairs_0=tmpMat else potentialPairs_0=rbind(potentialPairs_0, tmpMat)
;
}

}

#-----#
# E-M algorithm
#-----#

numPairs = numBlocks*blockSize^2;
pairsGammas = potentialPairs_0[,4:(4+numLinkVars-1)];
estParams = EMAlgorithm(numLinkVars=numLinkVars, blockSize=blockSize, pairsGammas=pairsGammas);

lambda = estParams$lambda;
m_probas = estParams$m_probas;
u_probas = estParams$u_probas;

m_gamma = rep(1,numPairs);
u_gamma = rep(1,numPairs);

for(k in 1:numLinkVars) {
  m_gamma = m_gamma*(m_probas[k]^pairsGammas[,k])*((1-m_probas[k])^(1-pairsGammas[,k]));
  u_gamma = u_gamma*(u_probas[k]^pairsGammas[,k])*((1-u_probas[k])^(1-pairsGammas[,k]));
}

w_gamma = log(m_gamma/u_gamma);
q_gamma = lambda*m_gamma/(lambda*m_gamma+(1-lambda)*u_gamma);

potentialPairs_0[,6+numLinkVars] = m_gamma;

```

```

potentialPairs_0[,7+numLinkVars] = u_gamma;
potentialPairs_0[,8+numLinkVars] = w_gamma;
potentialPairs_0[,9+numLinkVars] = q_gamma;
potentialPairs_0[,10+numLinkVars] = lambda;

result=determineThreshold(numLinkVars, estParams, target_fnr)

#print(result)

threshold=result$threshold

#print(list(threshold=threshold))

potentialPairs=cbind(potentialPairs_0, (potentialPairs_0[,8+numLinkVars])>=threshold))

#-----#
# Greedy linkage
#-----#

linkMatrices = list();

for(b in 1:numBlocks) {

  # Get the records in the block b in each file

  blockA=matrix(tableA[(tableA[,1]==b)], blockSize, ncols);
  blockB=matrix(tableB[(tableB[,1]==b)], blockSize, ncols);

  selectionSofar = c();

  for(r in 1:blockSize) {

    startIndex = (b-1)*blockSize^2+(r-1)*blockSize+1;
    endIndex = (b-1)*blockSize^2+r*blockSize;
    w_gamma = potentialPairs[startIndex:endIndex,8+numLinkVars];

    tmpMat0 = cbind(matrix(c(1:blockSize), blockSize, 1), c(w_gamma));
    tmpMat1 = matrix(tmpMat0[!(tmpMat0[,1] %in% selectionSofar),], ncol=2);

    max_w = max(tmpMat1[,2]);
    candidates = matrix(tmpMat1[(tmpMat1[,2]==max_w),], ncol=2);
    num_candidates = dim(candidates)[1];

    for(t in 1:num_candidates) {

      q = 1/(num_candidates-t+1);
      draw = rbinom(1,1,q);

      if(draw==1) {
        linkedRecidB = candidates[t,1];
        break;
      }
    }
  }
}

```

```

    }

    selectionSofar = c(selectionSofar, linkedRecidB);

    if(r==1) linkMatrix = matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1)
    else linkMatrix = cbind(linkMatrix, matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1));

  }

  linkMatrices[[b]] = linkMatrix;

}

#-----#
# Final output
#-----#

pairsData = list(
  numLinkVars = numLinkVars,
  mixingProportion = FPData$mixingProportion,
  lambda = lambda,
  m_probas = m_probas,
  u_probas = u_probas,
  linkMatrices = linkMatrices,
  potentialPairs = potentialPairs);

return(pairsData);

}

#-----#
# all gammas: a recursive function
#-----#

generateAllGammas = function (k)
{
  if(k==1) return(c(0,1))
  else {
    prevGammas = generateAllGammas(k-1);
    allGammas = rbind(matrix(rep(prevGammas,2),k-1,2^k),c(rep(0,2^(k-1)),rep(1,2^(k-1))));
    return(allGammas);
  }
}

#-----#
# E-M algorithm
#-----#

EMAlgorithm = function (numLinkVars, blockSize, pairsGammas)
{
  lambda = 1/blockSize;
  initMproba = runif(1,min=0.75, max=1.0);

```

```

initUproba = runif(1,min=0.0, max=0.25);

m_probas = matrix(rep(initMproba,numLinkVars),1,numLinkVars);
u_probas = matrix(rep(initUproba,numLinkVars),1,numLinkVars);

dFrame = as.data.frame(pairsGammas);
freqTable1 = as.data.frame(ftable(dFrame));
freqTable2 = freqTable1[(freqTable1$Freq>0),];

numProfiles=nrow(freqTable2);

for(col in 1:numLinkVars) {

  if(col==1) {
    profilesFreqs = matrix(c(freqTable2[,col])-1,numProfiles,1);
  }
  else {
    profilesFreqs = cbind(profilesFreqs, c(freqTable2[,col])-1);
  }
}

profilesFreqs = cbind(profilesFreqs, c(freqTable2$Freq), rep(1,numProfiles), rep(1,numProfiles),
  rep(0,numProfiles));

numIter = 100;

for(iter in 1:numIter) {

  estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

  profilesFreqs[,2+numLinkVars] = rep(1,numProfiles);
  profilesFreqs[,3+numLinkVars] = rep(1,numProfiles);

  for(k in 1:numLinkVars) {
    profilesFreqs[,2+numLinkVars] = profilesFreqs[,2+numLinkVars]*(m_probas[k]^profilesFreqs[,k])
      *((1-m_probas[k])^(1-profilesFreqs[,k]));
    profilesFreqs[,3+numLinkVars] = profilesFreqs[,3+numLinkVars]*(u_probas[k]^profilesFreqs[,k])
      *((1-u_probas[k])^(1-profilesFreqs[,k]));
  }

  profilesFreqs[,4+numLinkVars] = lambda*profilesFreqs[,2+numLinkVars]/(lambda*profilesFreqs[,2+
    numLinkVars]+(1-lambda)*profilesFreqs[,3+numLinkVars]);

  for(k in 1:numLinkVars) {
    m_probas[k] =sum(profilesFreqs[,k]*profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars])
      /sum(profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars]);
    u_probas[k] =sum(profilesFreqs[,k]*(1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+
      numLinkVars])/sum((1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+numLinkVars]);
  }

}

estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

```



```

    return(estParams);

}

#-----#
# determineThreshold:
#-----#

determineThreshold = function (numLinkVars, rlParams, target_fnr)
{
  allGammas<-generateAllGammas(numLinkVars)

  numProfiles<-2^numLinkVars

  lambda = rlParams$lambda;
  m_probas = rlParams$m_probas;
  u_probas = rlParams$u_probas;

  m_gamma = rep(1,numProfiles);
  u_gamma = rep(1,numProfiles);

  for(k in 1:numLinkVars) {
    m_gamma = m_gamma*(m_probas[k]^allGammas[k,])*((1-m_probas[k])^(1-allGammas[k,]));
    u_gamma = u_gamma*(u_probas[k]^allGammas[k,])*((1-u_probas[k])^(1-allGammas[k,]));
  }

  w_gamma = log(m_gamma/u_gamma);

  weight_order<-order(w_gamma);
  sum_m=m_gamma[weight_order[1]];

  t=1;
  while(sum_m<=target_fnr && t<numProfiles){
    t=t+1
    sum_m=sum_m+m_gamma[weight_order[t]]
  }
  if(t==1) threshold=w_gamma[weight_order[1]]
  else if(sum_m>target_fnr && t>1) threshold=w_gamma[weight_order[t-1]]
  else threshold=w_gamma[weight_order[t]]

  est_fnr=sum(m_gamma*(w_gamma<threshold))
  est_fpr=sum(u_gamma*(w_gamma>=threshold))

  result=list(threshold=threshold, est_fnr=est_fnr, est_fpr=est_fpr)

  return(result)
}

#-----#
# Performance measures
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse

```

```

#-----#

perfOne = function(beta_0, beta_1, all_estimates, scen, estMethod) {

  selectedscen=subset(all_estimates, (all_estimates[,1]==scen));
  selectedEstimates=subset(selectedscen, (selectedscen[,3]==estMethod));

  numRows=nrow(selectedEstimates);
  bias_beta0=round(100*mean(selectedEstimates[,4]-beta_0)/beta_0,3);
  mse_beta0=round(mean((selectedEstimates[,4]-beta_0)^2),6);
  var_beta0=round((sum((selectedEstimates[,4]-mean(selectedEstimates[,4]))^2)/(numRows-1)),6);

  bias_beta1=round(100*mean(selectedEstimates[,5]-beta_1)/beta_1,3);
  mse_beta1=round(mean((selectedEstimates[,5]-beta_1)^2),6);
  var_beta1=round((sum((selectedEstimates[,5]-mean(selectedEstimates[,5]))^2)/(numRows-1)),6);

  result=rbind(c(scen,estMethod,bias_beta0,var_beta0,mse_beta0),c(scen,estMethod,bias_beta1,
    var_beta1,mse_beta1));
  return(result)
}

#-----#
# Performance measure for all methods
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfAll = function(beta_0, beta_1, all_estimates, numScen) {

  for(scen in 1:numScen) {
    if(scen==1) allResults=perfOne(beta_0, beta_1, all_estimates, scen, 1)
    else allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates, scen, 1));
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates, scen, 4))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates, scen, 6))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates, scen, 7))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates, scen, 2))
  }
  return(allResults)
}

#-----#
# First and second moment of the match matrix
# a uniform random permutation
#-----#

momentsMatchMatrix = function (blockSize) {

  firstMoment = (1/blockSize)*matrix(rep(1,blockSize^2), blockSize, blockSize);

  for(i in 1:blockSize) {

    for(j in 1:blockSize) {

```

```

        E_m_ij_M = (1/(blockSize*(blockSize-1)))*matrix(rep(1,blockSize^2),blockSize,blockSize);
        E_m_ij_M[i,] = rep(0,blockSize);
        E_m_ij_M[,j] = rep(0,blockSize);
        E_m_ij_M[i,j] = 1/blockSize;

    if(j==1) currentBlockRow = E_m_ij_M else currentBlockRow = cbind(currentBlockRow, E_m_ij_M);

}

if(i==1) secondMoment = currentBlockRow else secondMoment = rbind(secondMoment, currentBlockRow)
;

}

moments = list(firstMoment=firstMoment, secondMoment=secondMoment);

return(moments);

}
#-----#
# Function: linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
# scoreOption:
#         0: complete data for linear and homoschedastic
#         1: naive BLUE for linear and homoschedastic
#-----#

linearScore = function(beta,pairs,minCmp,scoreOption) {

    numPairs=nrow(pairs);
    totalScore=0;
    beta_0=beta[1];
    beta_1=beta[2];

    for(r in 1:numPairs){
        x_hi=pairs[r,1];
        z_hj=pairs[r,2];
        m_hij=pairs[r,3];
        l_hij=(pairs[r,4]>=minCmp);
        eta_hi=beta_0+beta_1*x_hi;
        totalScore=totalScore+(scoreOption==0)*m_hij*(z_hj-eta_hi)^2;
        totalScore=totalScore+(scoreOption==1)*l_hij*(z_hj-eta_hi)^2;
    }
    return(totalScore);
}

#-----#
# Function: logistic score
# assume iid observations

```

```

# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l)
# scoreOption:
#       2: complete data for logistic
#       3: naive QL for logistic
#-----#

logitScore = function(beta, pairs, scoreOption) {

  numPairs=nrow(pairs);
  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  for(r in 1:numPairs){
    x_hi=pairs[r,1];
    z_hj=pairs[r,2];
    m_hij=pairs[r,3];
    l_hij=pairs[r,5];
    eta_hi=beta_0+beta_1*x_hi;
    mu_hi=exp(eta_hi)/(1+exp(eta_hi));
    totalScore=totalScore+(scoreOption==2)*m_hij*(z_hj-mu_hi)^2/(mu_hi*(1-mu_hi));
    totalScore=totalScore+(scoreOption==3)*l_hij*(z_hj-mu_hi)^2/(mu_hi*(1-mu_hi));
  }
  return(totalScore);
}

#-----#
# Function: survival score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l)
# scoreOption:
#       4: complete data for survival
#       5: naive QL for survival
#-----#

survivalScore = function(beta, pairs, followupTime, scoreOption) {

  numPairs=nrow(pairs);
  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  for(r in 1:numPairs){
    x_hi=pairs[r,1];
    z_hj=pairs[r,2];

```

```

        m_hij=pairs[r,3];
        l_hij=pairs[r,5];
        eta_hi=beta_0+beta_1*x_hi;
        f_ij=(z_hj<followupTime)*exp(eta_hi)*exp(-exp(eta_hi)*z_hj) + (z_hj>=followupTime)*exp(-
            exp(eta_hi)*z_hj);
        totalScore=totalScore+(scoreOption==4)*m_hij*log(f_ij);
        totalScore=totalScore+(scoreOption==5)*l_hij*log(f_ij);
    }
    return(totalScore);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
#         6: least squares PW
#-----#

PW1LinearScore1 = function(beta, pairs, numBlocks, blockSize, minCmp) {

    totalScore=0;
    beta_0=beta[1];
    beta_1=beta[2];

    for(b in 1:numBlocks) {
        block_pairs=subset(pairs, pairs[,6]==b)
        sum_x=sum(block_pairs[,1])

        subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
        numPairs=nrow(subset_block_pairs);

        if(numPairs>0){
            for(r in 1:numPairs){
                x_hi=subset_block_pairs[r,1];
                z_hj=subset_block_pairs[r,2];
                q_hij=subset_block_pairs[r,4];
                w_hij=q_hij*x_hi+(1-q_hij)*(sum_x-blockSize*x_hi)/(blockSize*(blockSize-1));
                eta_hij=beta_0+beta_1*w_hij;
                totalScore=totalScore+(z_hj-eta_hij)^2;
            }
        }
    }

    return(totalScore);
}

#-----#
# Function: Estimate sigma-sq for PW linear

```

```

# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
#
#-----#

PW1EstimateSigmasq = function(est_beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  total_numPairs=0;
  beta_0=est_beta[1];
  beta_1=est_beta[2];

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)

    sum_x=sum(block_pairs[,1])
    sum_x_sq=sum(block_pairs[,1]*block_pairs[,1])
    sum_xTx=matrix(c(blockSize*blockSize, sum_x, sum_x, sum_x_sq),2,2)

    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);
    total_numPairs=total_numPairs+numPairs;

    if (numPairs>0){
      for(r in 1:numPairs){
        x_hi=subset_block_pairs[r,1];
        z_hj=subset_block_pairs[r,2];
        q_hij=subset_block_pairs[r,4];
        w_hij=q_hij*x_hi+(1-q_hij)*(sum_x-blockSize*x_hi)/(blockSize*(blockSize-1));
        eta_hij=beta_0+beta_1*w_hij;

        totalScore=totalScore+(z_hj-eta_hij)^2-(q_hij*(beta_0+beta_1*x_hi)^2+t(est_beta)%*%((1-
          q_hij)/(blockSize-1))*(sum_xTx - blockSize*matrix(c(1, x_hi, x_hi, x_hi^2),2,2))/(
            blockSize*(blockSize-1))))%*%est_beta-eta_hij^2);
      }
    }
  }

  est_sigmaSq=totalScore/total_numPairs

  return(est_sigmaSq);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)

```

```

#           match status (m), cmp (q) and link status (l),
#           block id (b)
# scoreOption:
# 7: WLS PW with estimated variance
#-----#

PW1LinearScore2 = function(beta, est_beta, est_sigmaSq, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  est_beta_0=est_beta[1];
  est_beta_1=est_beta[2];

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)

    sum_x=sum(block_pairs[,1])
    sum_x_sq=sum(block_pairs[,1]*block_pairs[,1])
    sum_xTx=matrix(c(blockSize*blockSize, sum_x, sum_x, sum_x_sq),2,2)

    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);

    if(numPairs>0){
      for(r in 1:numPairs){
        x_hi=subset_block_pairs[r,1];
        z_hj=subset_block_pairs[r,2];
        q_hij=subset_block_pairs[r,4];
        w_hij=q_hij*x_hi+(1-q_hij)*(sum_x-blockSize*x_hi)/(blockSize*(blockSize-1));
        eta_hij=beta_0+beta_1*w_hij;
        est_eta_hij=est_beta_0+est_beta_1*w_hij;

        sigmaSq_hij=abs(est_sigmaSq+q_hij*(est_beta_0+est_beta_1*x_hi)^2+t(est_beta)%*%((1-q_hij)
          /(blockSize-1))*(sum_xTx - blockSize*matrix(c(1, x_hi, x_hi, x_hi^2),2,2))/(blockSize
            *(blockSize-1)))*%*%est_beta-est_eta_hij^2);

        totalScore=totalScore+(z_hj-eta_hij)^2/sigmaSq_hij;
      }
    }
  }

  return(totalScore);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)

```

```

#      match status (m), cmp (q) and link status (l),
#      block id (b)
# scoreOption:
# 8: PW composite likelihood
#-----#

PW1LinearScore3 = function(params, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=params[1];
  beta_1=params[2];
  sigmaSq=params[3];

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)
    block_etas=beta_0*c(1, blockSize)+beta_1*t(block_pairs[,1])
    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);
    if (numPairs>0){
      for(r in 1:numPairs){
        x_hi=subset_block_pairs[r,1];
        z_hj=subset_block_pairs[r,2];
        q_hij=subset_block_pairs[r,4];

        block_sum_pdf=sum(exp(-(z_hj*c(1, blockSize)-block_etas)^2/(2*sigmaSq))/(sqrt(2*pi*sigmaSq)
        ))
        f_hij=exp(-(z_hj-(beta_0+beta_1*x_hi))^2/sqrt(2*pi*sigmaSq))
        cond_pdf=q_hij*f_hij+(1-q_hij)*(block_sum_pdf-blockSize*f_hij)/(blockSize*(blockSize
        -1))
        totalScore=totalScore+log(cond_pdf);
      }
    }
  }

  return(totalScore);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#      match status (m), cmp (q) and link status (l),
#      block id (b)
# scoreOption:
#      9: PW2 least squares
#-----#

PW2LinearScore1 = function(beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

```



```

mean_x=mean(pairs[,1])

for(b in 1:numBlocks) {
  block_pairs=subset(pairs, pairs[,6]==b)
  subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
  numPairs=nrow(subset_block_pairs);

  if(numPairs>0){
    for(r in 1:numPairs){
      x_hi=subset_block_pairs[r,1];
      z_hj=subset_block_pairs[r,2];
      q_hij=subset_block_pairs[r,4];
      w_hij=q_hij*x_hi+(1-q_hij)*mean_x;
      eta_hij=beta_0+beta_1*w_hij;
      totalScore=totalScore+(z_hj-eta_hij)^2;
    }
  }
}

return(totalScore);
}

#-----#
# Function: Estimate sigma_sq for PW linear
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l),
#       block id (b)
#
#-----#

PW2EstimateSigmasq = function(est_beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  total_numPairs=0;
  beta_0=est_beta[1];
  beta_1=est_beta[2];

  mean_x=mean(pairs[,1])
  mean_x_sq=mean(pairs[,1]*pairs[,1])
  mean_xTx=matrix(c(blockSize*blockSize, mean_x, mean_x, mean_x_sq),2,2)

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)

    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);
    total_numPairs=total_numPairs+numPairs;

    if(numPairs>0){

```

```

    for(r in 1:numPairs){
      x_hi=subset_block_pairs[r,1];
      z_hj=subset_block_pairs[r,2];
      q_hij=subset_block_pairs[r,4];
      w_hij=q_hij*x_hi+(1-q_hij)*mean_x;
      eta_hij=beta_0+beta_1*w_hij;

      totalScore=totalScore+(z_hj-eta_hij)^2-(q_hij*(beta_0+beta_1*x_hi)^2+t(est_beta)%*%
        mean_xTx)%*%est_beta-eta_hij^2);
    }
  }
}

est_sigmaSq=totalScore/total_numPairs

return(est_sigmaSq);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
# 10: PW2 WLS with estimated variance
#-----#

PW2LinearScore2 = function(beta, est_beta, est_sigmaSq, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  est_beta_0=est_beta[1];
  est_beta_1=est_beta[2];

  mean_x=mean(pairs[,1])
  mean_x_sq=mean(pairs[,1]*pairs[,1])
  mean_xTx=matrix(c(blockSize*blockSize, mean_x, mean_x, mean_x_sq),2,2)

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)

    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);

    if(numPairs>0){
      for(r in 1:numPairs){
        x_hi=subset_block_pairs[r,1];
        z_hj=subset_block_pairs[r,2];

```

```

    q_hij=subset_block_pairs[r,4];
    w_hij=q_hij*x_hi+(1-q_hij)*mean_x;
    eta_hij=beta_0+beta_1*w_hij;
    est_eta_hij=est_beta_0+est_beta_1*w_hij;

    sigmaSq_hij=abs(est_sigmaSq+q_hij*(est_beta_0+est_beta_1*x_hi)^2+t(est_beta)%*%mean_xTx%*%
        est_beta-est_eta_hij^2);

    totalScore=totalScore+(z_hj-eta_hij)^2/sigmaSq_hij;
  }
}
}
return(totalScore);
}

#-----#
# Function: minimize score
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
# scoreType:
#
#         0: complete data for linear and homoschedastic
#         1: naive BLUE for linear and homoschedastic
#
#         2: complete data for logistic
#         3: naive QL for logistic
#
#         4: complete data for survival
#         5: naive QL for survival
#
# 6: PW LSE linear homoschedastic
# 7: PW WLS linear homoschedastic
#-----#

minimizeScore = function(initBeta=c(0,0), est_beta=c(0,0), est_sigmaSq=1.0, pairs, followupTime=0,
    numBlocks, blockSize, minCmp=0.0, scoreOption) {

  if(scoreOption==0 | scoreOption==1) result <- optim(initBeta, linearScore, gr=NULL, pairs=pairs,
    scoreOption=scoreOption, minCmp=minCmp, method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==2 | scoreOption==3) result <- optim(initBeta, logitScore, gr=NULL, pairs=
    pairs, scoreOption=scoreOption, method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==4 | scoreOption==5) result <- optim(initBeta, survivalScore, gr=NULL, pairs=
    pairs, followupTime=followupTime, scoreOption=scoreOption, method=c("BFGS"), control=list(
    fnscale=-1))
  else if(scoreOption==6) result <- optim(initBeta, PW1LinearScore1, gr=NULL, pairs=pairs,
    numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(
    fnscale=1))
  else if(scoreOption==7) result <- optim(initBeta, PW1LinearScore2, gr=NULL, est_beta=est_beta,
    est_sigmaSq=est_sigmaSq, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp,
    method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==8) {
    initParams=c(0.5,1.0,0.0625)

```

```

    result <- optim(initParams, PW1LinearScore3, gr=NULL, pairs=pairs, numBlocks=numBlocks,
      blockSize=blockSize, minCmp=minCmp, method="BFGS", control=list(fnscale=-1))
  }
  else if(scoreOption==9) result <- optim(initBeta, PW2LinearScore1, gr=NULL, pairs=pairs,
    numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(
      fnscale=1))
  else if(scoreOption==10) result <- optim(initBeta, PW2LinearScore2, gr=NULL, est_beta=est_beta,
    est_sigmaSq=est_sigmaSq, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp
    , method=c("BFGS"), control=list(fnscale=1))
  return(result);
}

#-----#
# Simulation parameters
#-----#

    numBlocks = 128;
    numRepetitions = 1000;
    numIter = 10;
    beta = matrix(c(0.5,1.0),2,1);

    scenarioList = list();
    ScenarioResults = list()

scenarioList[[1]] = list(blockSize=4,numLinkVars=8);
scenarioList[[2]] = list(blockSize=4,numLinkVars=8);
scenarioList[[3]] = list(blockSize=8,numLinkVars=8);

sink("output.txt");

#-----#
# Actual simulations
#-----#

#-----#
# All estimates
# for each row
# -scenario
# -iteration
# -cmp
# -method: 1-14
# -estimated beta_0
# -estimated beta_1
#-----#

for(scen in 1:1) {

    blockSize = scenarioList[[scen]]$blockSize;
    numLinkVars = scenarioList[[scen]]$numLinkVars;

    allGammas = t(generateAllGammas(numLinkVars));

```

```

for(t in 1:numRepetitions) {

  if(t%%10==0) {
    print(list(Iteration=t));
  }

  FPData = generateFinitePopulation(numBlocks=numBlocks, blockSize=blockSize, numLinkVars=
    numLinkVars);

  sigmasq=(FPData$sigma)^2;

  pairsData = generatePairs(FPData=FPData);
  popSize = FPData$popSize;
  blockSize = FPData$blockSize;
  numBlocks = FPData$numBlocks;
  xMat = cbind(rep(1, popSize), c(FPData$x));
  zMat = matrix(c(FPData$y), popSize, 1);

  momentsMatch = momentsMatchMatrix(blockSize=blockSize);
  E_Mh = momentsMatch$firstMoment;

  followupTime=0;

  # pairs: matrix with covariate (x), observed response (z)
  #      match status (m), cmp (q) and link status (l)

  potentialPairs = pairsData$potentialPairs;
  pairs = matrix(rep(0, 6*numBlocks*blockSize^2), numBlocks*blockSize^2, 6);
  pairs[,1] = potentialPairs[, (4+numLinkVars)];
  pairs[,2] = potentialPairs[, (5+numLinkVars)];
  pairs[,3] = potentialPairs[, (11+numLinkVars)];
  pairs[,4] = potentialPairs[, (9+numLinkVars)];
  pairs[,5] = potentialPairs[, (12+numLinkVars)];
  pairs[,6] = potentialPairs[, 1];

  initBeta=c(0,0);

  minCmp = 0.9;

  #-----#
  # Naive estimator
  # method 1
  #-----#

  scoreOption = 1;
  result = minimizeScore(initBeta=initBeta, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize,
    minCmp=minCmp, scoreOption=scoreOption);

  if(scen==1 && t==1) all_estimates=c(scen, t, 1, result$par)
  else all_estimates=rbind(all_estimates, c(scen, t, 1, result$par))
}

```

```

#-----#
# Complete data
# method 2
#-----#

scoreOption = 0;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp , scoreOption=scoreOption );
all_estimates=rbind(all_estimates , c(scen , t , 2 , result$par))

#-----#
# PW1 LSE
# method 3
#-----#

scoreOption = 6;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp , scoreOption=scoreOption );
est_beta =result$par;

all_estimates=rbind(all_estimates , c(scen , t , 3 , result$par))

#-----#
# PW1 WLSE
# method 4
#-----#

est_sigmaSq=PW1EstimateSigmasq(est_beta , pairs , numBlocks , blockSize , minCmp)
scoreOption = 7;
result = minimizeScore(initBeta=initBeta , est_beta=est_beta , est_sigmaSq=est_sigmaSq , pairs=pairs ,
    numBlocks=numBlocks , blockSize=blockSize , minCmp=minCmp , scoreOption=scoreOption );
all_estimates=rbind(all_estimates , c(scen , t , 4 , result$par))

#-----#
# PW2 LSE
# method 5
#-----#

scoreOption = 9;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp , scoreOption=scoreOption );
all_estimates=rbind(all_estimates , c(scen , t , 5 , result$par))
est_beta =result$par;

#-----#
# PW2 WLSE
# method 6
#-----#

est_sigmaSq=PW2EstimateSigmasq(est_beta , pairs , numBlocks , blockSize , minCmp)
scoreOption = 10;
result = minimizeScore(initBeta=initBeta , est_beta=est_beta , est_sigmaSq=est_sigmaSq , pairs=pairs ,
    numBlocks=numBlocks , blockSize=blockSize , minCmp=minCmp , scoreOption=scoreOption );

```

```

all_estimates=rbind(all_estimates,c(scen, t, 6, result$par))

#-----#
# Lahiri-Larsen
# method 7
#-----#

Sum1 = matrix(rep(0,2),2,1);
Sum2 = matrix(rep(0,4),2,2);

for(h in 1:numBlocks){

  startIndex = blockSize*(h-1)+1;
  endIndex = blockSize*h;
  Z_h = zMat[startIndex:endIndex];
  X_h = xMat[startIndex:endIndex,];
  W_h = t(E_Mh)%*%X_h;

  Sum1 = Sum1 + t(W_h)%*%Z_h;
  Sum2 = Sum2 + t(W_h)%*%W_h;
}

LLEstimate = solve(Sum2, Sum1);

all_estimates=rbind(all_estimates,c(scen, t, 7, LLEstimate))

}

write.csv(all_estimates, file = "all_estimates.csv")

numScen=1
allResults=perfAll(beta[1], beta[2], all_estimates, numScen)

write.csv(allResults, file = "results.csv")

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

}

sink();

```

B.1.2 Logistic regression

The following R code was used.

```

#-----#
#

```

```

# randomPermutation(blockSize=)
#
#-----#

randomPermutation = function (blockSize)
{
    u = runif(blockSize,0,1);
    sortedU = sort(u);

    #i=perm(j)

    permutationMatrix = matrix(rep(0,blockSize^2),blockSize,blockSize);

    for (j in 1:blockSize)

        for (i in 1:blockSize) permutationMatrix[i,j] = (u[i]==sortedU[j]);

    return(permutationMatrix);
}

#-----#
#
# generateFinitePopulation(numBlocks=, blockSize=, numLinkVars=)
#
#-----#

generateFinitePopulation = function (numBlocks,
                                     blockSize,
                                     numLinkVars) # all binary variables
{

    ALPHA = 0.5;
    BETA = 1.0;
    MEAN_X = 0.0;
    SIGMA_X = 1.0;
    #num_x_steps = 10;
    SIGMA = 1.0;
    P = 0.5;
    Q0 = 0.05;
    Q1 = 0.95;
    SHUFFLEPROBA = 1.0;

    #for low quality Q0=0.2, Q1=0.8
    #for medium quality Q0=0.1, Q1=0.9
    #for high quality Q0=0.05, Q1=0.95

    popSize = numBlocks*blockSize;

    blocks = c(t(matrix(rep(c(1:numBlocks),blockSize),numBlocks,blockSize)));

    mixingProportion = 1/blockSize;

```



```

uAgree = (P*Q1+(1-P)*Q0)^2+(1-(P*Q1+(1-P)*Q0))^2;
mAgree = P*(Q1^2+(1-Q1)^2)+(1-P)*((1-Q0)^2+Q0^2);

#uAgree = 1/4;
#mAgree = 1/2;

x = runif(popSize,min=-1,max=1);
#x = round(-(num_x_steps/2)+num_x_steps*runif(popSize,0,1),0)/(num_x_steps/2);
eta=ALPHA+BETA*x;
mu=exp(eta)/(1+exp(eta));
y=rbinom(popSize,1,mu);
#y = ALPHA+BETA*x+SIGMA*rnorm(popSize,0,1);

origLinkVars = matrix(rbinom(popSize*numLinkVars,1,P),popSize,numLinkVars);

recordedLinkVarsA = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars,1,Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars,1,Q1),popSize,numLinkVars);
recordedLinkVarsB = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars,1,Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars,1,Q1),popSize,numLinkVars);

blockids = c(t(matrix(rep(c(1:numBlocks),blockSize), numBlocks, blockSize)));

recidA = cbind(blockids, matrix(rep(matrix(c(1:blockSize), blockSize, 1),numBlocks),popSize,1));
#oRecidB = recidA;
recidB = recidA;

#-----#
# Apply a random permutation to B records
#-----#

# Shuffle the linkage variables and the responses
# within each block

#recidB = oRecidB;
oRecidB = recidB;

matchMatrices = list();

for(b in 1:numBlocks) {

  startIndex = (b-1)*blockSize+1;
  endIndex = b*blockSize;
  permMat = diag(rep(1,blockSize));

  shuffleYes = rbinom(1,1,SHUFFLEPROBA);

  if(shuffleYes) {
    permMat = randomPermutation(blockSize);
  }
}

```

```

        oRecidB[startIndex:endIndex,1:2] = permMat%%recidB[startIndex:endIndex
        ,1:2];
        recordedLinkVarsB[startIndex:endIndex,1:numLinkVars] = permMat%%recordedLinkVarsB[startIndex:
        endIndex,1:numLinkVars];
        y[startIndex:endIndex] = permMat%%y[startIndex:endIndex];
    }

    matchMatrices[[b]]= permMat;

}

FPData = list(
    numBlocks = numBlocks,
    blockSize = blockSize,
    popSize = popSize,
    numLinkVars = numLinkVars,
    blocks = blocks,
    recidA = recidA,
    oRecidB = oRecidB,
    recidB = recidB,
    matchMatrices = matchMatrices,
    origLinkVars = origLinkVars,
    recordedLinkVarsA = recordedLinkVarsA,
    recordedLinkVarsB = recordedLinkVarsB,
    shuffleProba = SHUFFLEPROBA,
    p = P,
    q0 = Q0,
    q1 = Q1,
    mixingProportion = mixingProportion,
    mAgree = mAgree,
    uAgree = uAgree,
    x = x,
    y = y,
    model = 'linear regression',
    alpha = ALPHA,
    beta = BETA,
    sigma = SIGMA,
    meanX = MEAN_X,
    sigmaX = SIGMA_X);

return(FPData);
}

#-----#
# compareLinkVars(v1, v2)
#-----#

compareLinkVars = function(v1, v2) {

    numLinkVars = dim(v1)[2];
    numPairs = dim(v1)[1]

    gammas = matrix(rep(0,numLinkVars*numPairs),numPairs,numLinkVars);

```

```

for(i in 1:numPairs)
  for(j in 1:numLinkVars) gammas[i,j] = (v1[i,j]==v2[i,j]);

return(gammas);
}

#-----#
# generatePairs(FPData)
#-----#

generatePairs = function(FPData)
{

  popSize      = FPData$popSize;
  numBlocks    = FPData$numBlocks;
  blockSize    = FPData$blockSize;
  numLinkVars  = FPData$numLinkVars;
  recidA       = FPData$recidA;
  recidB       = FPData$recidB;
  oRecidB      = FPData$oRecidB;
  target_fnr   = 0.05;

  mixingProportion = FPData$mixingProportion;

  mAgree = FPData$mAgree;
  uAgree = FPData$uAgree;

  shuffleProba = FPData$shuffleProba;

  ncols = 2+numLinkVars;

  #-----#
  # tableA: with ncols columns
  #       col 1: (perfect) blocking variable
  #       col 2 through col ncols-1: linkage variables
  #       col ncols: covariate x
  #
  # tableB: with ncols columns
  #       col 1: (perfect) blocking variable
  #       col 2 through col ncols-1: linkage variables
  #       col ncols: covariate y
  #-----#

  tableA = cbind(FPData$blocks, FPData$recordedLinkVarsA, FPData$x);
  tableB = cbind(FPData$blocks, FPData$recordedLinkVarsB, FPData$y);

  for(b in 1:numBlocks) {

    # Get the records in the block b in each file

    blockA=matrix(tableA[(tableA[,1]==b),],blockSize,ncols);
    blockB=matrix(tableB[(tableB[,1]==b),],blockSize,ncols);

```

```

# if (0) {
  startIndex_b = (b-1)*blockSize+1;
  endIndex_b = b*blockSize;

  oRecidB_b = matrix(rep(0,blockSize),blockSize,1);
  oRecidB_b = oRecidB[startIndex_b:endIndex_b,2];
#}

#print(list(oRecidB=oRecidB));

for(r in 1:blockSize) {

#-----#
# For t the block size build
#
#      x_r y_1
#      x_r y_2
#      .   .
#      .   .
#      .   .
#      x_r y_t
#-----#

  analyticalVars = cbind(matrix(rep(blockA[r,ncols],blockSize),blockSize,1), blockB[,ncols]);

#-----#
# For t the block size build
# v_j the vector of linkage
#      variables for record j
#
#      gamma(v_r, v_1)
#      gamma(v_r, v_2)
#      .   .
#      .   .
#      .   .
#      gamma(v_r, v_t)
#-----#

  gammas = compareLinkVars(t(matrix(rep(blockA[r,2:(ncols-1)],blockSize),numLinkVars, blockSize),
    blockB[,2:(ncols-1)]));
  tmpMat = cbind(rep(b,blockSize), rep(r,blockSize), c(1:blockSize), gammas, analyticalVars,
    matrix(rep(0,5*blockSize),blockSize,5), 1*(oRecidB_b==r));

  if(b==1 && r==1) potentialPairs_0=tmpMat else potentialPairs_0=rbind(potentialPairs_0, tmpMat)
  ;
}

}

#-----#

```

```

# E-M algorithm
#-----#

numPairs = numBlocks*blockSize^2;
pairsGammas = potentialPairs_0[,4:(4+numLinkVars-1)];
estParams = EMAlgorithm(numLinkVars=numLinkVars, blockSize=blockSize, pairsGammas=pairsGammas);

lambda = estParams$lambda;
m_probas = estParams$m_probas;
u_probas = estParams$u_probas;

m_gamma = rep(1,numPairs);
u_gamma = rep(1,numPairs);

for(k in 1:numLinkVars) {
  m_gamma = m_gamma*(m_probas[k]^pairsGammas[,k])*((1-m_probas[k])^(1-pairsGammas[,k]));
  u_gamma = u_gamma*(u_probas[k]^pairsGammas[,k])*((1-u_probas[k])^(1-pairsGammas[,k]));
}

w_gamma = log(m_gamma/u_gamma);
q_gamma = lambda*m_gamma/(lambda*m_gamma+(1-lambda)*u_gamma);

potentialPairs_0[,6+numLinkVars] = m_gamma;
potentialPairs_0[,7+numLinkVars] = u_gamma;
potentialPairs_0[,8+numLinkVars] = w_gamma;
potentialPairs_0[,9+numLinkVars] = q_gamma;
potentialPairs_0[,10+numLinkVars] = lambda;

result=determineThreshold(numLinkVars, estParams, target_fnr)

#print(result)

threshold=result$threshold

#print(list(threshold=threshold))

potentialPairs=cbind(potentialPairs_0, (potentialPairs_0[,8+numLinkVars])>=threshold))

#-----#
# Greedy linkage
#-----#

linkMatrices = list();

for(b in 1:numBlocks) {

  # Get the records in the block b in each file

  blockA=matrix(tableA[(tableA[,1]==b)], blockSize, ncols);
  blockB=matrix(tableB[(tableB[,1]==b)], blockSize, ncols);

  selectionSofar = c();

```

```

for(r in 1:blockSize) {

  startIndex = (b-1)*blockSize^2+(r-1)*blockSize+1;
  endIndex = (b-1)*blockSize^2+r*blockSize;
  w_gamma = potentialPairs[startIndex:endIndex,8+numLinkVars];

  tmpMat0 = cbind(matrix(c(1:blockSize),blockSize,1),c(w_gamma));
  tmpMat1 = matrix(tmpMat0[!(tmpMat0[,1] %in% selectionSofar),],ncol=2);

  max_w = max(tmpMat1[,2]);
  candidates = matrix(tmpMat1[(tmpMat1[,2]==max_w),],ncol=2);
  num_candidates = dim(candidates)[1];

  for(t in 1:num_candidates) {

    q = 1/(num_candidates-t+1);
    draw = rbinom(1,1,q);

    if(draw==1) {
      linkedRecidB = candidates[t,1];
      break;
    }

  }

  selectionSofar = c(selectionSofar, linkedRecidB);

  if(r==1) linkMatrix = matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1)
  else linkMatrix = cbind(linkMatrix, matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1));

}

linkMatrices[[b]] = linkMatrix;

}

#-----#
# Final output
#-----#

pairsData = list(
  numLinkVars = numLinkVars,
  mixingProportion = FPDData$mixingProportion,
  lambda = lambda,
  m_probas = m_probas,
  u_probas = u_probas,
  linkMatrices = linkMatrices,
  potentialPairs = potentialPairs);

return(pairsData);

}

```

```

#-----#
# all gammas: a recursive function
#-----#

generateAllGammas = function (k)
{
  if(k==1) return(c(0,1))
  else {
    prevGammas = generateAllGammas(k-1);
    allGammas = rbind(matrix(rep(prevGammas,2),k-1,2^k),c(rep(0,2^(k-1)),rep(1,2^(k-1))));
    return(allGammas);
  }
}

#-----#
# E-M algorithm
#-----#

EMAlgorithm = function (numLinkVars, blockSize, pairsGammas)
{
  lambda = 1/blockSize;
  initMproba = runif(1,min=0.75, max=1.0);
  initUproba = runif(1,min=0.0, max=0.25);

  m_probas = matrix(rep(initMproba,numLinkVars),1,numLinkVars);
  u_probas = matrix(rep(initUproba,numLinkVars),1,numLinkVars);

  dFrame = as.data.frame(pairsGammas);
  freqTable1 = as.data.frame(ftable(dFrame));
  freqTable2 = freqTable1[(freqTable1$Freq>0),];

  numProfiles=nrow(freqTable2);

  for(col in 1:numLinkVars) {

    if(col==1) {
      profilesFreqs = matrix(c(freqTable2[,col])-1,numProfiles,1);
    }
    else {
      profilesFreqs = cbind(profilesFreqs, c(freqTable2[,col])-1);
    }
  }

  profilesFreqs = cbind(profilesFreqs, c(freqTable2$Freq), rep(1,numProfiles), rep(1,numProfiles),
    rep(0,numProfiles));

  numIter = 100;

  for(iter in 1:numIter) {

    estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

```

```

profilesFreqs[,2+numLinkVars] = rep(1,numProfiles);
profilesFreqs[,3+numLinkVars] = rep(1,numProfiles);

for(k in 1:numLinkVars) {
  profilesFreqs[,2+numLinkVars] = profilesFreqs[,2+numLinkVars]*(m_probas[k]^profilesFreqs[,k])
  *((1-m_probas[k])^(1-profilesFreqs[,k]));
  profilesFreqs[,3+numLinkVars] = profilesFreqs[,3+numLinkVars]*(u_probas[k]^profilesFreqs[,k])
  *((1-u_probas[k])^(1-profilesFreqs[,k]));
}

profilesFreqs[,4+numLinkVars] = lambda*profilesFreqs[,2+numLinkVars]/(lambda*profilesFreqs[,2+
  numLinkVars]+(1-lambda)*profilesFreqs[,3+numLinkVars]);

for(k in 1:numLinkVars) {
  m_probas[k] =sum(profilesFreqs[,k]*profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars])
  /sum(profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars]);
  u_probas[k] =sum(profilesFreqs[,k]*(1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+
  numLinkVars])/sum((1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+numLinkVars]);
}

}

estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

return(estParams);

}

#-----#
# determineThreshold:
#-----#

determineThreshold = function (numLinkVars, rlParams, target_fnr)
{
  allGammas<-generateAllGammas(numLinkVars)

  numProfiles<-2^numLinkVars

  lambda = rlParams$lambda;
  m_probas = rlParams$m_probas;
  u_probas = rlParams$u_probas;

  m_gamma = rep(1,numProfiles);
  u_gamma = rep(1,numProfiles);

  for(k in 1:numLinkVars) {
    m_gamma = m_gamma*(m_probas[k]^allGammas[k,])*((1-m_probas[k])^(1-allGammas[k,]));
    u_gamma = u_gamma*(u_probas[k]^allGammas[k,])*((1-u_probas[k])^(1-allGammas[k,]));
  }

  w_gamma = log(m_gamma/u_gamma);

  weight_order<-order(w_gamma);

```



```

sum_m=m_gamma[ weight_order [1]];

t=1;
while (sum_m<=target_fnr && t<numProfiles){
  t=t+1
  sum_m=sum_m+m_gamma[ weight_order [t]]
}
if (t==1) threshold=w_gamma[ weight_order [1]]
else if (sum_m>target_fnr && t>1) threshold=w_gamma[ weight_order [t-1]]
else threshold=w_gamma[ weight_order [t]]

est_fnr=sum (m_gamma*(w_gamma<threshold))
est_fpr=sum (u_gamma*(w_gamma>=threshold))

result=list (threshold=threshold , est_fnr=est_fnr , est_fpr=est_fpr)

return (result)
}

#-----#
# Performance measures
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfOne = function (beta_0 , beta_1 , all_estimates , scen , estMethod) {

  selectedscen=subset ( all_estimates , ( all_estimates [,1]==scen));
  selectedEstimates=subset (selectedscen , (selectedscen [,3]==estMethod));

  numRows=nrow (selectedEstimates);
  bias_beta0=round (100*mean (selectedEstimates [,4] - beta_0)/beta_0 ,3);
  mse_beta0=round (mean ((selectedEstimates [,4] - beta_0)^2) ,6);
  var_beta0=round ((sum ((selectedEstimates [,4] - mean (selectedEstimates [,4]))^2)/(numRows-1)) ,6);

  bias_beta1=round (100*mean (selectedEstimates [,5] - beta_1)/beta_1 ,3);
  mse_beta1=round (mean ((selectedEstimates [,5] - beta_1)^2) ,6);
  var_beta1=round ((sum ((selectedEstimates [,5] - mean (selectedEstimates [,5]))^2)/(numRows-1)) ,6);

  result=rbind (c (scen ,estMethod ,bias_beta0 ,var_beta0 ,mse_beta0) ,c (scen ,estMethod ,bias_beta1 ,
    var_beta1 ,mse_beta1));
  return (result)
}

#-----#
# Performance measure for all methods
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfAll = function (beta_0 , beta_1 , all_estimates ,numScen) {

  for (scen in 1:numScen) {

```

```

    if(scen==1) allResults=perfOne(beta_0, beta_1, all_estimates,scen,1)
    else allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,1));
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,3))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,4))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,5))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,6))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,2))
  }

  for(t in 1:6) {
    if(t==1) finalResults=allResults[1,]
    else finalResults=rbind(finalResults, allResults[2*(t-1)+1,])
  }
  for(t in 1:6) finalResults=rbind(finalResults, allResults[2*t,]);

  return(finalResults)
}

#-----#
# First and second moment of the match matrix
# a uniform random permutation
#-----#

momentsMatchMatrix = function (blockSize) {

  firstMoment = (1/blockSize)*matrix(rep(1,blockSize^2), blockSize, blockSize);

  for(i in 1:blockSize) {

    for(j in 1:blockSize) {

      E_m_ij_M = (1/(blockSize*(blockSize-1)))*matrix(rep(1,blockSize^2),blockSize,blockSize);
      E_m_ij_M[i,] = rep(0,blockSize);
      E_m_ij_M[,j] = rep(0,blockSize);
      E_m_ij_M[i,j] = 1/blockSize;

      if(j==1) currentBlockRow = E_m_ij_M else currentBlockRow = cbind(currentBlockRow, E_m_ij_M);

    }

    if(i==1) secondMoment = currentBlockRow else secondMoment = rbind(secondMoment, currentBlockRow)
    ;

  }

  moments = list(firstMoment=firstMoment, secondMoment=secondMoment);

  return(moments);

}

#-----#
# Function: logistic score

```

```

# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
# scoreOption:
#         2: complete data for logistic
#         3: naive QL for logistic
#-----#

logitScore = function(beta, pairs, minCmp, scoreOption) {

  numPairs=nrow(pairs);
  beta_0=beta[1];
  beta_1=beta[2];

  total=c(0,0)

  for(r in 1:numPairs){
    x_hi=pairs[r,1];
    z_hj=pairs[r,2];
    m_hij=pairs[r,3];
    l_hij=(pairs[r,4]>=minCmp);
    eta_hi=beta_0+beta_1*x_hi;
    mu_hi=exp(eta_hi)/(1+exp(eta_hi));
    total=total+((scoreOption==0)*m_hij+(scoreOption==1)*l_hij)*(z_hj-mu_hi)*c(1,x_hi);
  }
  score=sum(total*total)

  return(score);
}

#-----#
# Function: Pairwise logit score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
#         6: least squares PW
#-----#

PW1LogitScore1 = function(beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)

```

```

        etas=beta_0+beta_1*block_pairs[,1]
        mus=exp(etas)/(1+exp(etas))
        sum_mus=sum(mus)

#       vars=mus*(1-mus)
#       sum_vars=sum(vars)

#       print(list(etas=etas, mus=mus, vars=vars))

subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
        numPairs=nrow(subset_block_pairs);

if (numPairs>0){
  for (r in 1:numPairs){
    x_hi=subset_block_pairs[r,1];
    z_hj=subset_block_pairs[r,2];
    q_hij=subset_block_pairs[r,4];
    eta_hi=beta_0+beta_1*x_hi
    mu_hi=exp(eta_hi)/(1+exp(eta_hi))
    mu_hij=q_hij*mu_hi+(1-q_hij)*(sum_mus-blockSize*mu_hi)/(blockSize*(blockSize-1));

    totalScore=totalScore+(z_hj-mu_hij)^2;
  }
}

return(totalScore);
}

#-----#
# Function: Pairwise logist score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
# 7: WLS PW
#-----#

PW1LogitScore2 = function(beta, est_beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  est_beta_0=est_beta[1];
  est_beta_1=est_beta[2];

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)

```

```

etas=beta_0+beta_1*block_pairs[,1]
mus=exp(etas)/(1+exp(etas))
sum_mus=sum(mus)

est_etas=est_beta_0+est_beta_1*block_pairs[,1]
est_mus=exp(est_etas)/(1+exp(est_etas))
est_sum_mus=sum(est_mus)

est_vars=exp(est_etas)/(1+exp(est_etas))^2
est_sum_vars=sum(est_vars)

subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
numPairs=nrow(subset_block_pairs);

if (numPairs>0){
  for(r in 1:numPairs){
    x_hi=subset_block_pairs[r,1];
    z_hj=subset_block_pairs[r,2];
    q_hij=subset_block_pairs[r,4];
    eta_hi=beta_0+beta_1*x_hi
    mu_hi=exp(eta_hi)/(1+exp(eta_hi))
    mu_hij=q_hij*mu_hi+(1-q_hij)*(sum_mus-blockSize*mu_hi)/(blockSize*(blockSize-1));

    est_eta_hi=est_beta_0+est_beta_1*x_hi
    est_mu_hi=exp(est_eta_hi)/(1+exp(est_eta_hi))
    est_var_hi=est_mu_hi*(1-est_mu_hi)
    est_mu_hij=q_hij*est_mu_hi+(1-q_hij)*(est_sum_mus-blockSize*est_mu_hi)/(blockSize*(
      blockSize-1));

    sigmaSq_hij=abs(q_hij*(est_mu_hi^2+est_var_hi)+(1-q_hij)*((est_sum_vars - blockSize*
      est_var_hi)+(sum(est_mus^2) - blockSize*est_mu_hi^2))/(blockSize*(blockSize-1))-
      est_mu_hij^2);

    totalScore=totalScore+(z_hj-mu_hij)^2/sigmaSq_hij;
  }
}

return(totalScore);
}

#-----#
# Function: Pairwise logit score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l),
#       block id (b)
# scoreOption:
#       9: PW2 least squares
#-----#

```

```

PW2LogitScore1 = function(beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  etas=beta_0+beta_1*pairs[,1]
  all_mus=exp(etas)/(1+exp(etas))
  mean_mus=mean(all_mus)

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)
    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);

    if(numPairs>0){
      for(r in 1:numPairs){
        x_hi=subset_block_pairs[r,1];
        z_hj=subset_block_pairs[r,2];
        q_hij=subset_block_pairs[r,4];
        eta_hi=beta_0+beta_1*x_hi
        mu_hi=exp(eta_hi)/(1+exp(eta_hi))
        mu_hij=q_hij*mu_hi+(1-q_hij)*mean_mus;
        totalScore=totalScore+(z_hj-mu_hij)^2;
      }
    }
  }
  return(totalScore);
}

#-----#
# Function: Pairwise logit score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
# 10: PW2 WLS
#-----#

PW2LogitScore2 = function(beta, est_beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  est_beta_0=est_beta[1];
  est_beta_1=est_beta[2];

  etas=beta_0+beta_1*pairs[,1]
  all_mus=exp(etas)/(1+exp(etas))

```

```

mean_mus=mean(all_mus)

    est_etas=est_beta_0+est_beta_1*pairs[,1]
    est_all_mus=exp(est_etas)/(1+exp(est_etas))
    est_mean_mus=mean(est_all_mus)
    est_mean_var=mean(est_all_mus*(1-est_all_mus))
    est_mean_musq=mean(est_all_mus^2)

for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)

    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);

    if(numPairs>0){
        for(r in 1:numPairs){
            x_hi=subset_block_pairs[r,1];
            z_hj=subset_block_pairs[r,2];
            q_hij=subset_block_pairs[r,4];
            eta_hi=beta_0+beta_1*x_hi
            mu_hi=exp(eta_hi)/(1+exp(eta_hi))
            mu_hij=q_hij*mu_hi+(1-q_hij)*mean_mus;

            est_eta_hi=est_beta_0+est_beta_1*x_hi
            est_mu_hi=exp(est_eta_hi)/(1+exp(est_eta_hi))
            est_var_hi=est_mu_hi*(1-est_mu_hi)
            est_mu_hij=q_hij*est_mu_hi+(1-q_hij)*est_mean_mus;

            sigmaSq_hij=abs(q_hij*(est_mu_hi^2+est_var_hi)+(1-q_hij)*(est_mean_var+est_mean_musq)-
                est_mu_hij^2);
            totalScore=totalScore+(z_hj-mu_hij)^2/sigmaSq_hij;
        }
    }
}
return(totalScore);
}

#-----#
# Function: minimize score
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l)
# scoreType:
#
#       0: complete data QL
#       1: naive QL
#
#       2: PW1 LSE
#       3: PW2 WLS
#
#       4: PW LSE
#       5: PW WLS
#-----#

```

```

minimizeScore = function(initBeta=c(0,0), est_beta=c(0,0), pairs, numBlocks, blockSize, minCmp
                        =0.0, scoreOption) {

  if(scoreOption==0 | scoreOption==1) result <- optim(initBeta, logitScore, gr=NULL, pairs=pairs,
    scoreOption=scoreOption, minCmp=minCmp, method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==2) result <- optim(initBeta, PW1LogitScore1, gr=NULL, pairs=pairs, numBlocks
    =numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==3) result <- optim(initBeta, PW1LogitScore2, gr=NULL, est_beta=est_beta,
    pairs=pairs, numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"),
    control=list(fnscale=1))
  else if(scoreOption==4) result <- optim(initBeta, PW2LogitScore1, gr=NULL, pairs=pairs,
    numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(
    fnscale=1))
  else if(scoreOption==5) result <- optim(initBeta, PW2LogitScore2, gr=NULL, est_beta=est_beta,
    pairs=pairs, numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"),
    control=list(fnscale=1))
  return(result);
}

#-----#
# Simulation parameters
#-----#

      numBlocks = 128;
      numRepetitions = 100;
      numIter = 10;
      beta = matrix(c(0.5,1.0),2,1);

      scenarioList = list();
      ScenarioResults = list()

      scenarioList[[1]]=list(blockSize=8,numLinkVars=8);
      scenarioList[[2]]=list(blockSize=4,numLinkVars=8);
      scenarioList[[3]]=list(blockSize=8,numLinkVars=8);

      sink("output.txt");

#-----#
# Actual simulations
#-----#

#-----#
# All estimates
# for each row
# -scenario
# -iteration
# -cmp
# -method: 1-14
# -estimated beta_0
# -estimated beta_1
#-----#

```



```

for(scen in 1:1) {

  blockSize = scenarioList[[scen]]$blockSize;
  numLinkVars = scenarioList[[scen]]$numLinkVars;

  allGammas = t(generateAllGammas(numLinkVars));

  for(t in 1:numRepetitions) {

    if(t%%5==0) {
      print(list(Iteration=t));
    }

    FPData = generateFinitePopulation(numBlocks=numBlocks, blockSize=blockSize, numLinkVars=
      numLinkVars);

    pairsData = generatePairs(FPData=FPData);
    popSize = FPData$popSize;
    blockSize = FPData$blockSize;
    numBlocks = FPData$numBlocks;
    xMat = cbind(rep(1,popSize),c(FPData$x));
    zMat = matrix(c(FPData$y),popSize,1);

    momentsMatch = momentsMatchMatrix(blockSize=blockSize);
    E_Mh = momentsMatch$firstMoment;

    # pairs: matrix with covariate (x), observed response (z)
    #          match status (m), cmp (q) and link status (l)

    potentialPairs = pairsData$potentialPairs;
    pairs = matrix(rep(0,6*numBlocks*blockSize^2),numBlocks*blockSize^2,6);
    pairs[,1] = potentialPairs[, (4+numLinkVars)];
    pairs[,2] = potentialPairs[, (5+numLinkVars)];
    pairs[,3] = potentialPairs[, (11+numLinkVars)];
    pairs[,4] = potentialPairs[, (9+numLinkVars)];
    pairs[,5] = potentialPairs[, (12+numLinkVars)];
    pairs[,6] = potentialPairs[,1];

    initBeta=beta;
    minCmp = 0.9;

    #-----#
    # Naive estimator
    # method 1
    #-----#

    scoreOption = 1;
    result = minimizeScore(initBeta=initBeta, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize,
      minCmp=minCmp, scoreOption=scoreOption);

    if(scen==1 && t==1) all_estimates=c(scen, t, 1, result$par)
    else all_estimates=rbind(all_estimates,c(scen, t, 1, result$par))
  }
}

```

```

#-----#
# Complete data
# method 2
#-----#

scoreOption = 0;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp , scoreOption=scoreOption );
all_estimates=rbind(all_estimates , c(scen , t , 2 , result$par))

#-----#
# PW1 LSE
# method 3
#-----#

scoreOption = 2;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp , scoreOption=scoreOption );
est_beta =result$par;
all_estimates=rbind(all_estimates , c(scen , t , 3 , result$par))

#-----#
# PW1 WLSE
# method 4
#-----#
#if(0){
scoreOption = 3;
result = minimizeScore(initBeta=initBeta , est_beta=est_beta , pairs=pairs , numBlocks=numBlocks ,
    blockSize=blockSize , minCmp=minCmp , scoreOption=scoreOption );
all_estimates=rbind(all_estimates , c(scen , t , 4 , result$par))

#-----#
# PW2 LSE
# method 5
#-----#

scoreOption = 4;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp , scoreOption=scoreOption );
all_estimates=rbind(all_estimates , c(scen , t , 5 , result$par))
est_beta =result$par;

#-----#
# PW2 WLSE
# method 6
#-----#

scoreOption = 5;
result = minimizeScore(initBeta=initBeta , est_beta=est_beta , pairs=pairs , numBlocks=numBlocks ,
    blockSize=blockSize , minCmp=minCmp , scoreOption=scoreOption );
all_estimates=rbind(all_estimates , c(scen , t , 6 , result$par))
#}
}

```

```

write.csv(all_estimates, file = "logit_all_estimates_k8_Nh8_cmp90.csv")

numScen=1
allResults=perfAll(beta[1], beta[2], all_estimates, numScen)

write.csv(allResults, file = "logitResults_k8_Nh8_cmp90.csv")

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

}

sink();

```

B.1.3 Survival model

The following R code was used.

```

#-----#
# randomPermutation(blockSize=)
#-----#

randomPermutation = function (blockSize)
{
    u = runif(blockSize,0,1);
    sortedU = sort(u);

    #i=perm(j)

    permutationMatrix = matrix(rep(0,blockSize^2),blockSize,blockSize);

    for (j in 1:blockSize)

        for (i in 1:blockSize) permutationMatrix[i,j] = (u[i]==sortedU[j]);

    return(permutationMatrix);
}

#-----#
# generateFinitePopulation(numBlocks=, blockSize=, numLinkVars=)
#-----#

generateFinitePopulation = function (numBlocks,
                                     blockSize,
                                     numLinkVars) # all binary variables
{

```

```

ALPHA = 0.5;
BETA  = 1.0;
P      = 0.5;
Q0     = 0.05;
Q1     = 0.95;
SHUFFLEPROBA = 1.0;
followupTime = 2.0;

#for low quality Q0=0.2, Q1=0.8
#for medium quality Q0=0.1, Q1=0.9
#for high quality Q0=0.05, Q1=0.95

popSize = numBlocks*blockSize;

blocks = c(t(matrix(rep(c(1:numBlocks), blockSize), numBlocks, blockSize)));

mixingProportion = 1/blockSize;

uAgree = (P*Q1+(1-P)*Q0)^2+(1-(P*Q1+(1-P)*Q0))^2;
mAgree = P*(Q1^2+(1-Q1)^2)+(1-P)*((1-Q0)^2+Q0^2);

#uAgree = 1/4;
#mAgree = 1/2;

x = 2*rbinom(popSize, 1, 0.5);
eta = ALPHA+BETA*x;
survivalTimes = -exp(-eta)*log(1-runif(popSize, 0, 1));
censored = (survivalTimes>followupTime)
y = followupTime*censored+(1-censored)*survivalTimes

origLinkVars = matrix(rbinom(popSize*numLinkVars, 1, P), popSize, numLinkVars);

recordedLinkVarsA = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars, 1, Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars, 1, Q1), popSize, numLinkVars);
recordedLinkVarsB = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars, 1, Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars, 1, Q1), popSize, numLinkVars);

blockids = c(t(matrix(rep(c(1:numBlocks), blockSize), numBlocks, blockSize)));

recidA = cbind(blockids, matrix(rep(matrix(c(1:blockSize), blockSize, 1), numBlocks), popSize, 1));
recidB = recidA;

#-----#
# Apply a random permutation to B records
#-----#

# Shuffle the linkage variables and the responses
# within each block

```

```

oRecidB = recidB;

matchMatrices = list();

for(b in 1:numBlocks) {

  startIndex = (b-1)*blockSize+1;
  endIndex = b*blockSize;
  permMat = diag(rep(1,blockSize));

  shuffleYes = rbinom(1,1,SHUFFLEPROBA);

  if(shuffleYes) {
    permMat = randomPermutation(blockSize);
    oRecidB[startIndex:endIndex,1:2] = permMat%%recidB[startIndex:endIndex,1:2];
    recordedLinkVarsB[startIndex:endIndex,1:numLinkVars] = permMat%%recordedLinkVarsB[startIndex:endIndex,1:numLinkVars];

    survivalTimes[startIndex:endIndex] = permMat%%survivalTimes[startIndex:endIndex];
    censored[startIndex:endIndex] = permMat%%censored[startIndex:endIndex];
    y[startIndex:endIndex] = permMat%%y[startIndex:endIndex];
  }

  matchMatrices[[b]] = permMat;
}

FPData = list(
  numBlocks = numBlocks,
  blockSize = blockSize,
  popSize = popSize,
  numLinkVars = numLinkVars,
  blocks = blocks,
  recidA = recidA,
  oRecidB = oRecidB,
  recidB = recidB,
  matchMatrices = matchMatrices,
  origLinkVars = origLinkVars,
  recordedLinkVarsA = recordedLinkVarsA,
  recordedLinkVarsB = recordedLinkVarsB,
  shuffleProba = SHUFFLEPROBA,
  p = P,
  q0 = Q0,
  q1 = Q1,
  mixingProportion = mixingProportion,
  mAgree = mAgree,
  uAgree = uAgree,
  x = x,
  y = y,
  survivalTimes = survivalTimes,

```

```

        followupTime = followupTime,
        censored = censored,
        model = 'Survival PHM',
        alpha = ALPHA,
        beta = BETA);

    return(FPData);
}

#-----#
#
# compareLinkVars(v1, v2)
#
#-----#

compareLinkVars = function(v1, v2) {

    numLinkVars = dim(v1)[2];
    numPairs = dim(v1)[1]

    gammas = matrix(rep(0,numLinkVars*numPairs),numPairs,numLinkVars);

    for(i in 1:numPairs)
        for(j in 1:numLinkVars) gammas[i,j] = (v1[i,j]==v2[i,j]);

    return(gammas);
}

#-----#
#
# generatePairs(FPData)
#
#-----#

generatePairs = function(FPData)
{

    popSize      = FPData$popSize;
    numBlocks    = FPData$numBlocks;
    blockSize    = FPData$blockSize;
    numLinkVars  = FPData$numLinkVars;
    recidA       = FPData$recidA;
    recidB       = FPData$recidB;
    oRecidB      = FPData$oRecidB;
    target_fnr   = 0.05;
    #censored    = FPData$censored;
    mixingProportion = FPData$mixingProportion;

    mAgree = FPData$mAgree;
    uAgree = FPData$uAgree;

    shuffleProba = FPData$shuffleProba;

```

```

ncols = 2+numLinkVars;

#-----#
# tableA: with ncols columns
#       col 1: (perfect) blocking variable
#       col 2 through col ncols-1: linkage variables
#       col ncols: covariate x
#
# tableB: with ncols columns
#       col 1: (perfect) blocking variable
#       col 2 through col ncols-1: linkage variables
#       col ncols: covariate y
#-----#

tableA = cbind(FPData$blocks, FPData$recordedLinkVarsA, FPData$x);
tableB = cbind(FPData$blocks, FPData$recordedLinkVarsB, FPData$y, FPData$censored);

for(b in 1:numBlocks) {

  # Get the records in the block b in each file

  blockA=matrix(tableA[(tableA[,1]==b),], blockSize, ncols);
  blockB=matrix(tableB[(tableB[,1]==b),], blockSize, ncols+1);

  startIndex.b=(b-1)*blockSize+1;
  endIndex.b=b*blockSize;

  oRecidB.b=matrix(rep(0, blockSize), blockSize, 1);
  oRecidB.b[oRecidB[startIndex.b:endIndex.b, 2]];

  #print(list(oRecidB=oRecidB));

  for(r in 1:blockSize) {

    #-----#
    # For t the block size build
    #
    #       x_r y_1
    #       x_r y_2
    #       .   .
    #       .   .
    #       .   .
    #       x_r y_t
    #-----#

    analyticalVars = cbind(matrix(rep(blockA[r, ncols], blockSize), blockSize, 1), blockB[, ncols:(
      ncols+1)]);

    #-----#
    # For t the block size build
    # v_j the vector of linkage
    #       variables for record j

```

```

#
#      gamma(v_r, v_l)
#      gamma(v_r, v_2)
#      .      .
#      .      .
#      .      .
#      gamma(v_r, v_t)
#-----#

gammas = compareLinkVars(t(matrix(rep(blockA[r, 2:(ncols-1)], blockSize), numLinkVars, blockSize)
    ), blockB[, 2:(ncols-1)]);

#-----#
# 1) block no
# 2) recid A
# 3) recid B
# 4) to (3+numLinkVars) gamma1
#    through gamma-K
#
# (4+numLinkVars) to
# (6+numLinkVars) x, y and censored
#
# (7+numLinkVars) to
# (11+numLinkVars) m- and u- probas
# linkage weight, cmp and lambda
#
# 12+numLinkVars match status
#-----#

tmpMat = cbind(rep(b, blockSize), rep(r, blockSize), c(1:blockSize), gammas, analyticalVars,
    matrix(rep(0, 5*blockSize), blockSize, 5), 1*(oRecidB.b==r));

if(b==1 && r==1) potentialPairs_0=tmpMat else potentialPairs_0=rbind(potentialPairs_0, tmpMat)
;
}

}

#-----#
# E-M algorithm
#-----#

numPairs = numBlocks*blockSize^2;
pairsGammas = potentialPairs_0[, 4:(4+numLinkVars-1)];
estParams = EMAlgorithm(numLinkVars=numLinkVars, blockSize=blockSize, pairsGammas=pairsGammas);

lambda = estParams$lambda;
m-probas = estParams$m-probas;
u-probas = estParams$u-probas;

m_gamma = rep(1, numPairs);
u_gamma = rep(1, numPairs);

```



```

for(k in 1:numLinkVars) {
  m_gamma = m_gamma*(m_probas[k]^pairsGammas[,k])*((1-m_probas[k])^(1-pairsGammas[,k]));
  u_gamma = u_gamma*(u_probas[k]^pairsGammas[,k])*((1-u_probas[k])^(1-pairsGammas[,k]));
}

w_gamma = log(m_gamma/u_gamma);
q_gamma = lambda*m_gamma/(lambda*m_gamma+(1-lambda)*u_gamma);

potentialPairs_0[,7+numLinkVars] = m_gamma;
potentialPairs_0[,8+numLinkVars] = u_gamma;
potentialPairs_0[,9+numLinkVars] = w_gamma;
potentialPairs_0[,10+numLinkVars] = q_gamma;
potentialPairs_0[,11+numLinkVars] = lambda;

result=determineThreshold(numLinkVars, estParams, target_fnr)
threshold=result$threshold

potentialPairs=cbind(potentialPairs_0, (potentialPairs_0[,8+numLinkVars])>=threshold))

#-----#
# Greedy linkage
#-----#

linkMatrices = list();

for(b in 1:numBlocks) {

  # Get the records in the block b in each file

  blockA=matrix(tableA[(tableA[,1]==b)], blockSize, ncols);
  blockB=matrix(tableB[(tableB[,1]==b)], blockSize, ncols+1);

  selectionSofar = c();

  for(r in 1:blockSize) {

    startIndex = (b-1)*blockSize^2+(r-1)*blockSize+1;
    endIndex = (b-1)*blockSize^2+r*blockSize;
    w_gamma = potentialPairs[startIndex:endIndex,8+numLinkVars];

    tmpMat0 = cbind(matrix(c(1:blockSize), blockSize, 1), c(w_gamma));
    tmpMat1 = matrix(tmpMat0[!(tmpMat0[,1] %in% selectionSofar),], ncol=2);

    max_w = max(tmpMat1[,2]);
    candidates = matrix(tmpMat1[(tmpMat1[,2]==max_w),], ncol=2);
    num_candidates = dim(candidates)[1];

    for(t in 1:num_candidates) {

      q = 1/(num_candidates-t+1);
      draw = rbinom(1,1,q);

      if(draw==1) {

```

```

        linkedRecidB = candidates[t,1];
        break;
    }

}

selectionSofar = c(selectionSofar, linkedRecidB);

if(r==1) linkMatrix = matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1)
else linkMatrix = cbind(linkMatrix, matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1));

}

linkMatrices[[b]] = linkMatrix;

}

#-----#
# Final output
#-----#

pairsData = list(
    numLinkVars = numLinkVars,
    mixingProportion = FPData$mixingProportion,
    lambda = lambda,
    m_probas = m_probas,
    u_probas = u_probas,
    linkMatrices = linkMatrices,
    potentialPairs = potentialPairs);

return(pairsData);

}

#-----#
# all gammas: a recursive function
#-----#

generateAllGammas = function (k)
{
    if(k==1) return(c(0,1))
    else {
        prevGammas = generateAllGammas(k-1);
        allGammas = rbind(matrix(rep(prevGammas,2),k-1,2^k),c(rep(0,2^(k-1)),rep(1,2^(k-1))));
        return(allGammas);
    }
}

#-----#
# E-M algorithm
#-----#

EMAlgorithm = function (numLinkVars, blockSize, pairsGammas)

```

```

{

    lambda = 1/blockSize;
    initMproba = runif(1,min=0.75, max=1.0);
    initUproba = runif(1,min=0.0, max=0.25);

    m_probas = matrix(rep(initMproba, numLinkVars), 1, numLinkVars);
    u_probas = matrix(rep(initUproba, numLinkVars), 1, numLinkVars);

    dFrame = as.data.frame(pairsGammas);
    freqTable1 = as.data.frame(ftable(dFrame));
    freqTable2 = freqTable1[(freqTable1$Freq > 0),];

    numProfiles = nrow(freqTable2);

    for(col in 1:numLinkVars) {

        if(col==1) {
            profilesFreqs = matrix(c(freqTable2[, col]) - 1, numProfiles, 1);
        }
        else {
            profilesFreqs = cbind(profilesFreqs, c(freqTable2[, col]) - 1);
        }
    }

    profilesFreqs = cbind(profilesFreqs, c(freqTable2$Freq), rep(1, numProfiles), rep(1, numProfiles),
        rep(0, numProfiles));

    numIter = 100;

    for(iter in 1:numIter) {

        estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

        profilesFreqs[, 2+numLinkVars] = rep(1, numProfiles);
        profilesFreqs[, 3+numLinkVars] = rep(1, numProfiles);

        for(k in 1:numLinkVars) {
            profilesFreqs[, 2+numLinkVars] = profilesFreqs[, 2+numLinkVars] * (m_probas[k]^profilesFreqs[, k])
                * ((1-m_probas[k])^(1-profilesFreqs[, k]));
            profilesFreqs[, 3+numLinkVars] = profilesFreqs[, 3+numLinkVars] * (u_probas[k]^profilesFreqs[, k])
                * ((1-u_probas[k])^(1-profilesFreqs[, k]));
        }

        profilesFreqs[, 4+numLinkVars] = lambda*profilesFreqs[, 2+numLinkVars]/(lambda*profilesFreqs[, 2+
            numLinkVars] + (1-lambda)*profilesFreqs[, 3+numLinkVars]);

        for(k in 1:numLinkVars) {
            m_probas[k] = sum(profilesFreqs[, k]*profilesFreqs[, 4+numLinkVars]*profilesFreqs[, 1+numLinkVars])
                /sum(profilesFreqs[, 4+numLinkVars]*profilesFreqs[, 1+numLinkVars]);
            u_probas[k] = sum(profilesFreqs[, k]*(1-profilesFreqs[, 4+numLinkVars])*profilesFreqs[, 1+
                numLinkVars])/sum((1-profilesFreqs[, 4+numLinkVars])*profilesFreqs[, 1+numLinkVars]);
        }
    }
}

```

```

}

estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

return(estParams);

}

#-----#
# determineThreshold:
#-----#

determineThreshold = function (numLinkVars, rlParams, target_fnr)
{
  allGammas<-generateAllGammas(numLinkVars)

  numProfiles<-2^numLinkVars

  lambda = rlParams$lambda;
  m_probas = rlParams$m_probas;
  u_probas = rlParams$u_probas;

  m_gamma = rep(1,numProfiles);
  u_gamma = rep(1,numProfiles);

  for(k in 1:numLinkVars) {
    m_gamma = m_gamma*(m_probas[k]^allGammas[k,])*((1-m_probas[k])^(1-allGammas[k,]));
    u_gamma = u_gamma*(u_probas[k]^allGammas[k,])*((1-u_probas[k])^(1-allGammas[k,]));
  }

  w_gamma = log(m_gamma/u_gamma);

  weight_order<-order(w_gamma);
  sum_m=m_gamma[weight_order[1]];

  t=1;
  while(sum_m<=target_fnr && t<numProfiles){
    t=t+1
    sum_m=sum_m+m_gamma[weight_order[t]]
  }
  if(t==1) threshold=w_gamma[weight_order[1]]
  else if(sum_m>target_fnr && t>1) threshold=w_gamma[weight_order[t-1]]
  else threshold=w_gamma[weight_order[t]]

  est_fnr=sum(m_gamma*(w_gamma<threshold))
  est_fpr=sum(u_gamma*(w_gamma>=threshold))

  result=list(threshold=threshold, est_fnr=est_fnr, est_fpr=est_fpr)

  return(result)
}

```

```

#-----#
# Performance measures
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfOne = function(beta_0, beta_1, all_estimates, scen, estMethod) {

  selectedscen=subset(all_estimates, (all_estimates[,1]==scen));
  selectedEstimates=subset(selectedscen, (selectedscen[,3]==estMethod));

  numRows=nrow(selectedEstimates);
  bias_beta0=round(100*mean(selectedEstimates[,4] - beta_0)/beta_0,3);
  mse_beta0=round(mean((selectedEstimates[,4] - beta_0)^2),6);
  var_beta0=round((sum((selectedEstimates[,4] - mean(selectedEstimates[,4]))^2)/(numRows-1)),6);

  bias_beta1=round(100*mean(selectedEstimates[,5] - beta_1)/beta_1,3);
  mse_beta1=round(mean((selectedEstimates[,5] - beta_1)^2),6);
  var_beta1=round((sum((selectedEstimates[,5] - mean(selectedEstimates[,5]))^2)/(numRows-1)),6);

  result=rbind(c(scen,estMethod,bias_beta0,var_beta0,mse_beta0),c(scen,estMethod,bias_beta1,
    var_beta1,mse_beta1));
  return(result)
}

#-----#
# Performance measure for all methods
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfAll = function(beta_0, beta_1, all_estimates,numScen) {

  for(scen in 1:numScen) {
    if(scen==1) allResults=perfOne(beta_0, beta_1, all_estimates,scen,1)
    else allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,1));
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,3))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,4))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,2))
  }

  for(t in 1:4) {
    if(t==1) finalResults=allResults[1,]
    else finalResults=rbind(finalResults, allResults[2*(t-1)+1,])
  }
  for(t in 1:4) finalResults=rbind(finalResults, allResults[2*t,]);

  return(finalResults)
}

#-----#
# Function: survival score
# assume iid observations

```

```

# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l)
# scoreOption:
#       4: complete data for survival
#       5: naive QL for survival
#-----#

survivalScore = function(beta, pairs, minCmp, scoreOption) {

  numPairs=nrow(pairs);
  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  for(r in 1:numPairs){
    x_hi=pairs[r,1];
    z_hj=pairs[r,2];
    censored_hj=pairs[r,7];
    m_hij=pairs[r,3];
    l_hij=(pairs[r,4]>=minCmp);
    eta_hi=beta_0+beta_1*x_hi;
    log_f_ij=((scoreOption==0)*m_hij+(scoreOption==1)*l_hij)*((1-censored_hj)*(eta_hi-exp(eta_hi)
      *z_hj) + censored_hj*(-exp(eta_hi)*z_hj));
    totalScore=totalScore+log_f_ij;
  }
  return(totalScore);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l),
#       block id (b)
# scoreOption:
#       6: least squares PW
#-----#

PW1SurvivalScore = function(beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)
    xs=block_pairs[,1]
    etas=beta_0+beta_1*xs
  }
}

```

```

subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
numPairs=nrow(subset_block_pairs);

if(numPairs>0){
  for(r in 1:numPairs){
    x_hi=subset_block_pairs[r,1];
    z_hj=subset_block_pairs[r,2];
    censored_hj=subset_block_pairs[r,7];
    q_hij=subset_block_pairs[r,4];
    eta_hi=beta_0+beta_1*x_hi;
    f_ij=(1-censored_hj)*exp(eta_hi)*exp(-exp(eta_hi)*z_hj) + censored_hj*exp(-exp(
      eta_hi)*z_hj)
    all_fs=(1-censored_hj)*exp(etas)*exp(-exp(etas)*z_hj) + censored_hj*exp(-exp(etas)*
      z_hj)
    avg_f_other=(sum(all_fs)-blockSize*f_ij)/(blockSize*(blockSize-1))
    totalScore=totalScore+log(q_hij*f_ij+(1-q_hij)*avg_f_other)
  }
}
return(totalScore);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l),
#       block id (b)
# scoreOption:
#       9: PW2 least squares
#-----#

PW2SurvivalScore = function(beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  xs=pairs[,1]
  etas=beta_0+beta_1*xs

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)
    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);

    if(numPairs>0){
      for(r in 1:numPairs){
        x_hi=subset_block_pairs[r,1];
        z_hj=subset_block_pairs[r,2];
        censored_hj=subset_block_pairs[r,7];

```

```

    q_hij=subset_block_pairs[r,4];
    eta_hi=beta_0+beta_1*x_hi;
    f_ij=(1-censored_hj)*exp(eta_hi)*exp(-exp(eta_hi)*z_hj) + censored_hj*exp(-exp(eta_hi)*
        z_hj)

    all_fs=(1-censored_hj)*exp(etas)*exp(-exp(etas)*z_hj) + censored_hj*exp(-exp(etas)*z_hj)
    avg_f=mean(all_fs)
    totalScore=totalScore+log(q_hij*f_ij+(1-q_hij)*avg_f)
  }
}
}
return(totalScore);
}

#-----#
# Function: minimize score
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l)
# scoreType:
#
#       0: complete data for linear and homoschedastic
#       1: naive BLUE for linear and homoschedastic
#
#       2: complete data for logistic
#       3: naive QL for logistic
#
#       4: complete data for survival
#       5: naive QL for survival
#
# 6: PW LSE linear homoschedastic
# 7: PW WLS linear homoschedastic
#-----#

maximizeScore = function(initBeta=c(0,0), pairs, followupTime=0, numBlocks, blockSize, minCmp=0.0,
    scoreOption) {

  if(scoreOption==0 | scoreOption==1) result <- optim(initBeta, survivalScore, gr=NULL, pairs=pairs
    , minCmp=minCmp, scoreOption=scoreOption, method=c("BFGS"), control=list(fnscale=-1))
  else if(scoreOption==2) result <- optim(initBeta, PW1SurvivalScore, gr=NULL, pairs=pairs,
    numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(
    fnscale=-1))
  else if(scoreOption==3) result <- optim(initBeta, PW2SurvivalScore, gr=NULL, pairs=pairs,
    numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(
    fnscale=-1))
  return(result);
}

#-----#
# Simulation parameters
#-----#

numBlocks = 128;

```



```

numRepetitions = 1000;
numIter = 10;
beta = matrix(c(0.5,1.0),2,1);

scenarioList = list();
ScenarioResults = list()

scenarioList[[1]] = list(blockSize=4,numLinkVars=8);
scenarioList[[2]] = list(blockSize=4,numLinkVars=8);
scenarioList[[3]] = list(blockSize=8,numLinkVars=8);

sink("output.txt");

#-----#
# Actual simulations
#-----#

#-----#
# All estimates
# for each row
# -scenario
# -iteration
# -cmp
# -method: 1-14
# -estimated beta_0
# -estimated beta_1
#-----#

for(scen in 1:1) {

  blockSize = scenarioList[[scen]]$blockSize;
  numLinkVars = scenarioList[[scen]]$numLinkVars;

  allGammas = t(generateAllGammas(numLinkVars));

  for(t in 1:numRepetitions) {

    if(t%%10==0) {
      print(list(Iteration=t));
    }

    FPData = generateFinitePopulation(numBlocks=numBlocks, blockSize=blockSize, numLinkVars=
      numLinkVars);

    #print(list(y=FPData$y))

    #-----#

    pairsData = generatePairs(FPData=FPData);
    popSize = FPData$popSize;
    blockSize = FPData$blockSize;
    numBlocks = FPData$numBlocks;

```

```

# pairs: matrix with covariate (x), observed response (z)
#           match status (m), cmp (q) and link status (l)
#           block (b), censored (c)

potentialPairs = pairsData$potentialPairs;
  pairs = matrix(rep(0,7*numBlocks*blockSize^2),numBlocks*blockSize^2,7);
pairs[,1] = potentialPairs[, (4+numLinkVars)];
pairs[,2] = potentialPairs[, (5+numLinkVars)];
pairs[,3] = potentialPairs[, (12+numLinkVars)];
pairs[,4] = potentialPairs[, (10+numLinkVars)];
pairs[,5] = potentialPairs[, (13+numLinkVars)];
pairs[,6] = potentialPairs[, 1];
pairs[,7] = potentialPairs[, (6+numLinkVars)];

initBeta=beta+c(rnorm(2,0,0.25));

minCmp = 0.9;

#-----#
# Naive estimator
# method 1
#-----#
scoreOption = 1;
result = maximizeScore(initBeta=initBeta, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize,
  minCmp=minCmp, scoreOption=scoreOption);
  if(scen==1 && t==1) all_estimates=c(scen, t, 1, result$par)
else all_estimates=rbind(all_estimates, c(scen, t, 1, result$par))

#-----#
# Complete data
# method 2
#-----#
scoreOption = 0;
result = maximizeScore(initBeta=initBeta, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize,
  minCmp=minCmp, scoreOption=scoreOption);
  all_estimates=rbind(all_estimates, c(scen, t, 2, result$par))

#-----#
# PW1
# method 3
#-----#
scoreOption = 2;
result = maximizeScore(initBeta=initBeta, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize,
  minCmp=minCmp, scoreOption=scoreOption);
all_estimates=rbind(all_estimates, c(scen, t, 3, result$par))

#-----#
# PW2
# method 4
#-----#
scoreOption = 3;
result = maximizeScore(initBeta=initBeta, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize,
  minCmp=minCmp, scoreOption=scoreOption);

```

```

    all_estimates=rbind(all_estimates,c(scen, t, 4, result$par))
  }

write.csv(all_estimates, file = "survival-all-estimates-k8-Nh4.csv")

numScen=1
allResults=perfAll(beta[1], beta[2], all_estimates, numScen)

write.csv(allResults, file = "survivalResults-k8-Nh4.csv")

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

#}
#-----#

}

sink();

```

B.2 Chapter 4

B.2.1 Linear regression

The following R code was used.

```

#-----#
#
# randomPermutation(blockSize=)
#
#-----#

randomPermutation = function (blockSize)
{
    u = runif(blockSize,0,1);
    sortedU = sort(u);

    permutationMatrix = matrix(rep(0,blockSize^2),blockSize,blockSize);

    for (j in 1:blockSize)
        for (i in 1:blockSize) permutationMatrix[i,j] = (u[i]==sortedU[j]);

    return(permutationMatrix);
}

```

```

#-----#
#
# generateFinitePopulation(numBlocks=, blockSize=, numLinkVars=)
#
#-----#

generateFinitePopulation = function (numBlocks,
                                     blockSize,
                                     numLinkVars) # all binary variables
{

  ALPHA = 0.5;
  BETA = 1.0;
  beta_missing_y_0=-2.0;
  beta_missing_y_1=1.0;
  beta_missing_x_0=-2.0;
  beta_missing_x_1=1.0;
  MEAN_X = 0.0;
  SIGMA_X = 1.0;
  SIGMA = 0.7;
  P = 0.5;
  Q0 = 0.05;
  Q1 = 0.95;
  SHUFFLEPROBA = 1.0;

  #for low quality Q0=0.2, Q1=0.8
  #for medium quality Q0=0.1, Q1=0.9
  #for high quality Q0=0.05, Q1=0.95

  popSize = numBlocks*blockSize;

  blocks = c(t(matrix(rep(c(1:numBlocks),blockSize),numBlocks,blockSize)));

  mixingProportion = 1/blockSize;

  uAgree = (P*Q1+(1-P)*Q0)^2+(1-(P*Q1+(1-P)*Q0))^2;
  mAgree = P*(Q1^2+(1-Q1)^2)+(1-P)*((1-Q0)^2+Q0^2);

  #uAgree = 1/4;
  #mAgree = 1/2;

  x = rnorm(popSize,MEAN_X, SIGMA_X);
  y = ALPHA+BETA*x+SIGMA*rnorm(popSize,0,1);

  eta_missing_x=beta_missing_x_0+beta_missing_x_1*x;

  eta_missing_y=beta_missing_y_0+beta_missing_y_1*x;

  p_missing_x=exp(eta_missing_x)/(1+exp(eta_missing_x));

  p_missing_y=exp(eta_missing_y)/(1+exp(eta_missing_y));

```

```

missing_x=rbinom(popSize,1,p_missing_x);
missing_y=rbinom(popSize,1,p_missing_y);

origLinkVars = matrix(rbinom(popSize*numLinkVars,1,P),popSize,numLinkVars);

recordedLinkVarsA = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars,1,Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars,1,Q1),popSize,numLinkVars);
recordedLinkVarsB = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars,1,Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars,1,Q1),popSize,numLinkVars);

blockids = c(t(matrix(rep(c(1:numBlocks),blockSize), numBlocks, blockSize)));

recidA = cbind(blockids, matrix(rep(matrix(c(1:blockSize), blockSize, 1),numBlocks),popSize,1));
recidB = recidA;

#-----#
# Apply a random permutation to B records
#-----#

# Shuffle the linkage variables and the responses
# within each block
oRecidB = recidB;

matchMatrices = list();

for(b in 1:numBlocks) {

  startIndex = (b-1)*blockSize+1;
  endIndex = b*blockSize;
  permMat = diag(rep(1,blockSize));

  shuffleYes = rbinom(1,1,SHUFFLEPROBA);

  if(shuffleYes) {
                                permMat = randomPermutation(blockSize);
    oRecidB[startIndex:endIndex,1:2] = permMat%%recidB[startIndex:endIndex
,1:2];
    recordedLinkVarsB[startIndex:endIndex,1:numLinkVars] = permMat%%recordedLinkVarsB[startIndex:
endIndex,1:numLinkVars];
                                y[startIndex:endIndex] = permMat%%y[startIndex:endIndex];
                                missing_y[startIndex:endIndex] = permMat%%missing_y[startIndex:endIndex
];
  }

  matchMatrices[[b]]= permMat;

}

FPData = list(      numBlocks = numBlocks,
                    blockSize = blockSize,

```

```

        popSize = popSize ,
        numLinkVars = numLinkVars ,
        blocks = blocks ,
        recidA = recidA ,
        oRecidB = oRecidB ,
        recidB = recidB ,
        matchMatrices = matchMatrices ,
        origLinkVars = origLinkVars ,
        recordedLinkVarsA = recordedLinkVarsA ,
        recordedLinkVarsB = recordedLinkVarsB ,
        shuffleProba = SHUFFLEPROBA,
        p = P,
        q0 = Q0,
        q1 = Q1,
        mixingProportion = mixingProportion ,
        mAgree = mAgree ,
        uAgree = uAgree ,
        x = x ,
        y = y ,
        missing_x = missing_x ,
        missing_y = missing_y ,
        p.missing_x=p.missing_x ,
        p.missing_y=p.missing_y ,
        model = 'linear regression ' ,
        alpha = ALPHA,
        beta = BETA,
        sigma = SIGMA,
        meanX = MEAN_X,
        sigmaX = SIGMA_X) ;

    return (FPData);
}

#-----#
#
# compareLinkVars(v1, v2)
#
#-----#

compareLinkVars = function(v1, v2) {

    numLinkVars = dim(v1)[2];
    numPairs = dim(v1)[1]

    gammas = matrix(rep(0,numLinkVars*numPairs),numPairs,numLinkVars);

    for(i in 1:numPairs)
        for(j in 1:numLinkVars) gammas[i,j] = (v1[i,j]==v2[i,j]);

    return(gammas);
}

```

```

#-----#
#
# generatePairs(FPData)
#
#-----#

generatePairs = function(FPData)
{

  popSize      = FPData$popSize;
  numBlocks    = FPData$numBlocks;
  blockSize    = FPData$blockSize;
  numLinkVars  = FPData$numLinkVars;
  recidA       = FPData$recidA;
  recidB       = FPData$recidB;
  oRecidB      = FPData$oRecidB;
  target_fnr   = 0.05;
  #missing_y   = FPData$missing-y;

  mixingProportion = FPData$mixingProportion;

  mAgree = FPData$mAgree;
  uAgree = FPData$uAgree;

  shuffleProba = FPData$shuffleProba;

  ncols      = 2+numLinkVars;
  ncols_A    = 5+numLinkVars;
  ncols_B    = 3+numLinkVars;

  #-----#
  # tableA: with ncols columns
  #       col 1: (perfect) blocking variable
  #       col 2 through col ncols-1: linkage variables
  #       col ncols: covariate x
  #
  # tableB: with ncols columns
  #       col 1: (perfect) blocking variable
  #       col 2 through col ncols-1: linkage variables
  #       col ncols: covariate y
  #-----#

  tableA = cbind(FPData$blocks, FPData$recordedLinkVarsA, FPData$x, FPData$missing_x,
    FPData$p-missing_x, FPData$p-missing-y);
  tableB = cbind(FPData$blocks, FPData$recordedLinkVarsB, FPData$y, FPData$missing-y);

  for(b in 1:numBlocks) {

    # Get the records in the block b in each file

    blockA=matrix(tableA[(tableA[,1]==b),], blockSize, ncols_A);
    blockB=matrix(tableB[(tableB[,1]==b),], blockSize, ncols_B);
  }
}

```

```

startIndex_b=(b-1)*blockSize+1;
endIndex_b=b*blockSize;

oRecidB_b=matrix(rep(0,blockSize),blockSize,1);
oRecidB_b=oRecidB[startIndex_b:endIndex_b,2];

for(r in 1:blockSize) {

#-----#
# For t the block size build
#
#      x_r y_1
#      x_r y_2
#      .   .
#      .   .
#      .   .
#      x_r y_t
#-----#

analyticalVars = cbind(t(matrix(rep(blockA[r,(ncols_A-3):ncols_A],blockSize),4,blockSize)),
      blockB[, (ncols_B-1):ncols_B]);

#-----#
# For t the block size build
# v_j the vector of linkage
#      variables for record j
#
#      gamma(v_r, v_1)
#      gamma(v_r, v_2)
#      .   .
#      .   .
#      .   .
#      gamma(v_r, v_t)
#-----#

gammas = compareLinkVars(t(matrix(rep(blockA[r,2:(ncols-1)],blockSize),numLinkVars, blockSize)
), blockB[,2:(ncols-1)]);
tmpMat = cbind(rep(b,blockSize), rep(r,blockSize), c(1:blockSize), gammas, analyticalVars,
      matrix(rep(0,5*blockSize),blockSize,5), 1*(oRecidB_b==r));

if(b==1 && r==1) potentialPairs_0=tmpMat else potentialPairs_0=rbind(potentialPairs_0, tmpMat)
;
}

}

#print(list(potentialPairs_0=potentialPairs_0))

#-----#
# E-M algorithm
#-----#

```



```

numPairs = numBlocks*blockSize^2;
pairsGammas = potentialPairs_0[,4:(4+numLinkVars-1)];
estParams = EMAlgorithm(numLinkVars=numLinkVars, blockSize=blockSize, pairsGammas=pairsGammas);

lambda = estParams$lambda;
m_probas = estParams$m_probas;
u_probas = estParams$u_probas;

m_gamma = rep(1,numPairs);
u_gamma = rep(1,numPairs);

for(k in 1:numLinkVars) {
  m_gamma = m_gamma*(m_probas[k]^pairsGammas[,k])*((1-m_probas[k])^(1-pairsGammas[,k]));
  u_gamma = u_gamma*(u_probas[k]^pairsGammas[,k])*((1-u_probas[k])^(1-pairsGammas[,k]));
}

w_gamma = log(m_gamma/u_gamma);
q_gamma = lambda*m_gamma/(lambda*m_gamma+(1-lambda)*u_gamma);

potentialPairs_0[,10+numLinkVars] = m_gamma;
potentialPairs_0[,11+numLinkVars] = u_gamma;
potentialPairs_0[,12+numLinkVars] = w_gamma;
potentialPairs_0[,13+numLinkVars] = q_gamma;
potentialPairs_0[,14+numLinkVars] = lambda;

result=determineThreshold(numLinkVars, estParams, target_fnr)

threshold=result$threshold

potentialPairs=cbind(potentialPairs_0, (potentialPairs_0[,12+numLinkVars])>=threshold))

#print(list(ncol=ncol(potentialPairs)))

#-----#
# Greedy linkage
#-----#

linkMatrices = list();

for(b in 1:numBlocks) {

  # Get the records in the block b in each file

  blockA=matrix(tableA[(tableA[,1]==b)], blockSize, ncols_A);
  blockB=matrix(tableB[(tableB[,1]==b)], blockSize, ncols_B);

  selectionSofar = c();

  for(r in 1:blockSize) {

    startIndex = (b-1)*blockSize^2+(r-1)*blockSize+1;
    endIndex = (b-1)*blockSize^2+r*blockSize;

```

```

w_gamma = potentialPairs[startIndex:endIndex,8+numLinkVars];

tmpMat0 = cbind(matrix(c(1:blockSize),blockSize,1),c(w_gamma));
tmpMat1 = matrix(tmpMat0[!(tmpMat0[,1] %in% selectionSofar),],ncol=2);

max_w = max(tmpMat1[,2]);
candidates = matrix(tmpMat1[(tmpMat1[,2]==max_w),],ncol=2);
num_candidates = dim(candidates)[1];

for(t in 1:num_candidates) {

  q = 1/(num_candidates-t+1);
  draw = rbinom(1,1,q);

  if(draw==1) {
    linkedRecidB = candidates[t,1];
    break;
  }

}

selectionSofar = c(selectionSofar, linkedRecidB);

if(r==1) linkMatrix = matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1)
else linkMatrix = cbind(linkMatrix, matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1));

}

linkMatrices[[b]] = linkMatrix;

}

#-----#
# Final output
#-----#

pairsData = list(
  numLinkVars = numLinkVars,
  mixingProportion = FPData$mixingProportion,
  lambda = lambda,
  m_probas = m_probas,
  u_probas = u_probas,
  linkMatrices = linkMatrices,
  potentialPairs = potentialPairs);

return(pairsData);

}

#-----#
# all gammas: a recursive function
#-----#

```

```

generateAllGammas = function (k)
{
  if (k==1) return(c(0,1))
  else {
    prevGammas = generateAllGammas(k-1);
    allGammas = rbind(matrix(rep(prevGammas,2),k-1,2^k),c(rep(0,2^(k-1)),rep(1,2^(k-1))));
    return(allGammas);
  }
}

#-----#
# E-M algorithm
#-----#

EMAlgorithm = function (numLinkVars, blockSize, pairsGammas)
{
  lambda = 1/blockSize;
  initMproba = runif(1,min=0.75, max=1.0);
  initUproba = runif(1,min=0.0, max=0.25);

  m_probas = matrix(rep(initMproba,numLinkVars),1,numLinkVars);
  u_probas = matrix(rep(initUproba,numLinkVars),1,numLinkVars);

  dFrame = as.data.frame(pairsGammas);
  freqTable1 = as.data.frame(ftable(dFrame));
  freqTable2 = freqTable1[(freqTable1$Freq>0),];

  numProfiles=nrow(freqTable2);

  for(col in 1:numLinkVars) {

    if(col==1) {
      profilesFreqs = matrix(c(freqTable2[,col])-1,numProfiles,1);
    }
    else {
      profilesFreqs = cbind(profilesFreqs, c(freqTable2[,col])-1);
    }
  }

  profilesFreqs = cbind(profilesFreqs, c(freqTable2$Freq), rep(1,numProfiles), rep(1,numProfiles),
    rep(0,numProfiles));

  numIter = 100;

  for(iter in 1:numIter) {

    estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

    profilesFreqs[,2+numLinkVars] = rep(1,numProfiles);
    profilesFreqs[,3+numLinkVars] = rep(1,numProfiles);

    for(k in 1:numLinkVars) {

```

```

    profilesFreqs[,2+numLinkVars] = profilesFreqs[,2+numLinkVars]*(m_probas[k]^profilesFreqs[,k])
    *((1-m_probas[k])^(1-profilesFreqs[,k]));
    profilesFreqs[,3+numLinkVars] = profilesFreqs[,3+numLinkVars]*(u_probas[k]^profilesFreqs[,k])
    *((1-u_probas[k])^(1-profilesFreqs[,k]));
}

profilesFreqs[,4+numLinkVars] = lambda*profilesFreqs[,2+numLinkVars]/(lambda*profilesFreqs[,2+
numLinkVars]+(1-lambda)*profilesFreqs[,3+numLinkVars]);

for(k in 1:numLinkVars) {
  m_probas[k] =sum(profilesFreqs[,k]*profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars])
    /sum(profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars]);
  u_probas[k] =sum(profilesFreqs[,k]*(1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+
numLinkVars])/sum((1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+numLinkVars]);
}

}

estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

return(estParams);

}

#-----#
# determineThreshold:
#-----#

determineThreshold = function (numLinkVars, rlParams, target_fnr)
{
  allGammas<-generateAllGammas(numLinkVars)

  numProfiles<-2^numLinkVars

  lambda = rlParams$lambda;
  m_probas = rlParams$m_probas;
  u_probas = rlParams$u_probas;

  m_gamma = rep(1,numProfiles);
  u_gamma = rep(1,numProfiles);

  for(k in 1:numLinkVars) {
    m_gamma = m_gamma*(m_probas[k]^allGammas[k,])*((1-m_probas[k])^(1-allGammas[k,]));
    u_gamma = u_gamma*(u_probas[k]^allGammas[k,])*((1-u_probas[k])^(1-allGammas[k,]));
  }

  w_gamma = log(m_gamma/u_gamma);

  weight_order<-order(w_gamma);
  sum_m=m_gamma[weight_order[1]];

  t=1;
  while(sum_m<=target_fnr && t<numProfiles){

```

```

    t=t+1
    sum_m=sum_m+m_gamma[ weight_order [ t ]]
  }
  if (t==1) threshold=w_gamma[ weight_order [ 1 ]]
  else if (sum_m>target_fnr && t>1) threshold=w_gamma[ weight_order [ t - 1 ]]
  else threshold=w_gamma[ weight_order [ t ]]

  est_fnr=sum(m_gamma*(w_gamma<threshold))
  est_fpr=sum(u_gamma*(w_gamma>=threshold))

  result=list(threshold=threshold, est_fnr=est_fnr, est_fpr=est_fpr)

  return(result)
}

#-----#
# Performance measures
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfOne = function(beta_0, beta_1, all_estimates, scen, estMethod) {

  selectedscen=subset(all_estimates, (all_estimates[,1]==scen));
  selectedEstimates=subset(selectedscen, (selectedscen[,3]==estMethod));

  numRows=nrow(selectedEstimates);
  bias_beta0=round(100*mean(selectedEstimates[,4] - beta_0)/beta_0,3);
  mse_beta0=round(mean((selectedEstimates[,4] - beta_0)^2),6);
  var_beta0=round((sum((selectedEstimates[,4] - mean(selectedEstimates[,4]))^2)/(numRows-1)),6);

  bias_beta1=round(100*mean(selectedEstimates[,5] - beta_1)/beta_1,3);
  mse_beta1=round(mean((selectedEstimates[,5] - beta_1)^2),6);
  var_beta1=round((sum((selectedEstimates[,5] - mean(selectedEstimates[,5]))^2)/(numRows-1)),6);

  result=rbind(c(scen,estMethod,bias_beta0,var_beta0,mse_beta0),c(scen,estMethod,bias_beta1,
    var_beta1,mse_beta1));
  return(result)
}

#-----#
# Performance measure for all methods
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfAll = function(beta_0, beta_1, all_estimates, numScen) {

  for(scen in 1:numScen) {
    if(scen==1) allResults=perfOne(beta_0, beta_1, all_estimates,scen,1)
    else allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,1));
    #allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,3))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,4))
  }
}

```

```

    #allResults=rbind(allResults , perfOne(beta_0 , beta_1 , all_estimates ,scen,5))
    allResults=rbind(allResults , perfOne(beta_0 , beta_1 , all_estimates ,scen,6))
    allResults=rbind(allResults , perfOne(beta_0 , beta_1 , all_estimates ,scen,2))
  }

  for(t in 1:4) {
    if(t==1) finalResults=allResults[1,]
    else finalResults=rbind(finalResults , allResults[2*(t-1)+1,])
  }
  for(t in 1:4) finalResults=rbind(finalResults , allResults[2*t,]);

  return(finalResults)
}

#-----#
# Function: linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
# scoreOption:
#         0: complete data for linear and homoschedastic
#         1: naive BLUE for linear and homoschedastic
# y is missing at random. Ignore the observations where y
# is missing
#-----#

linearScore = function(beta , pairs ,minCmp,scoreOption) {

  total=c(0,0);
  beta_0=beta[1];
  beta_1=beta[2];

  pairs_with_nomissing_x=subset(pairs , pairs[,7]==0)
  pairs_with_nomissing=subset(pairs_with_nomissing_x , pairs_with_nomissing_x[,8]==0)

  numPairs=nrow(pairs_with_nomissing);

  for(r in 1:numPairs){
    x_hi=pairs_with_nomissing[r,1];
    z_hj=pairs_with_nomissing[r,2];
    m_hij=pairs_with_nomissing[r,3];
    l_hij=(pairs_with_nomissing[r,4]>=minCmp);
    eta_hi=beta_0+beta_1*x_hi;
    total=total+((scoreOption==0)*m_hij+(scoreOption==1)*l_hij)*(z_hj-eta_hi)*c(1,x_hi);
  }
  score=sum(total*total)
  return(score);
}

#-----#
# Function: Pairwise linear score

```

```

# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
#         6: least squares PW
#-----#

PW1LinearScore1 = function(beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
  pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

  all_mus=beta_0+beta_1*pairs[,1]
  all_p_missing_y=pairs[,10]
  all_p_missing_x=pairs[,9]
  all_missing_x=pairs[,7]

  mean_1=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)*all_mus/(1-all_p_missing_x))
  mean_2=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)/(1-all_p_missing_x))
  mean_3=mean((1-all_missing_x)*all_p_missing_x/(1-all_p_missing_x))

  for(b in 1:numBlocks) {

    block_pairs=subset(pairs, pairs[,6]==b)
    n_h=blockSize-sum(subset(pairs, pairs[,6]==b)[,7])/blockSize

    block_pairs_nomissing=subset(pairs_with_nomissing, pairs_with_nomissing[,6]==b)
    block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing, block_pairs_nomissing[,4]>=
      minCmp);

    numPairs=nrow(block_pairs_nomissing_above_cmp);

    mus=beta_0+beta_1*block_pairs[,1]
    p_missing_y=block_pairs[,10]
    nonmissing_x=1-block_pairs[,7]

    bloc_total_1=sum(nonmissing_x*(1-p_missing_y)*mus)
    bloc_total_2=sum(nonmissing_x*(1-p_missing_y))

    if (numPairs>0){
      for(r in 1:numPairs){
        mu_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
        p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
        z_hj=block_pairs_nomissing_above_cmp[r,2]
        q_hij=block_pairs_nomissing_above_cmp[r,4]

        numerator_term_1_hij=(1-p_missing_y_hi)*mu_hi

```

```

        numerator_term_2_hij=(bloc_total_1-blockSize*numerator_term_1_hij)/(blockSize*(blockSize
        -1))
        numerator_term_3_hij=((blockSize-n_h)/(blockSize-1))*mean_1/mean_3
        numerator_hij=q_hij*numerator_term_1_hij+(1-q_hij)*(numerator_term_2_hij+
        numerator_term_3_hij)

        denominator_term_1_hij=1-p_missing_y_hi
        denominator_term_2_hij=(bloc_total_2-blockSize*denominator_term_1_hij)/(blockSize*(
        blockSize-1))
        denominator_term_3_hij=((blockSize-n_h)/(blockSize-1))*mean_2/mean_3
        denominator_hij=q_hij*denominator_term_1_hij+(1-q_hij)*(denominator_term_2_hij+
        denominator_term_3_hij)

        cond_mean_zhj=numerator_hij/denominator_hij

        totalScore=totalScore+(z_hj-cond_mean_zhj)^2;
    }
}
}

return(totalScore);
}

#-----#
# Function: Estimate sigma_sq for PW linear
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
#
#-----#

PW1EstimateSigmasq = function(beta, pairs, numBlocks, blockSize, minCmp) {

    totalScore=0
    total_numPairs=0

    beta_0=beta[1]
    beta_1=beta[2]

    pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
    pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

    all_mus=beta_0+beta_1*pairs[,1]
    all_p_missing_y=pairs[,10]
    all_p_missing_x=pairs[,9]
    all_missing_x=pairs[,7]

    mean_1=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)*all_mus/(1-all_p_missing_x))
    mean_2=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)/(1-all_p_missing_x))
    mean_3=mean((1-all_missing_x)*all_p_missing_x/(1-all_p_missing_x))

```



```

mean_4=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)*all_mus^2/(1-all_p_missing_x))

for(b in 1:numBlocks) {

  block_pairs=subset(pairs , pairs[,6]==b)
  n_h=blockSize-sum(subset(pairs , pairs[,6]==b)[,7])/blockSize

  block_pairs_nomissing=subset(pairs_with_nomissing , pairs_with_nomissing[,6]==b)
  block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing , block_pairs_nomissing[,4]>=
    minCmp);

  numPairs=nrow(block_pairs_nomissing_above_cmp)

  total_numPairs=total_numPairs+numPairs

  mus=beta_0+beta_1*block_pairs[,1]
  p_missing_y=block_pairs[,10]
  nonmissing_x=1-block_pairs[,7]

  bloc_total_1=sum(nonmissing_x*(1-p_missing_y)*mus)
  bloc_total_2=sum(nonmissing_x*(1-p_missing_y))
  bloc_total_3=sum(nonmissing_x*(1-p_missing_y)*mus^2)

#-----#

if (numPairs>0){

  for(r in 1:numPairs){

    #-----#
    # cond_mean_zhj
    #-----#

    mu_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
    p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
    z_hj=block_pairs_nomissing_above_cmp[r,2]
    q_hij=block_pairs_nomissing_above_cmp[r,4]

    numerator_term_1_hij=(1-p_missing_y_hi)*mu_hi
    numerator_term_2_hij=(bloc_total_1-blockSize*numerator_term_1_hij)/(blockSize*(blockSize
      -1))
    numerator_term_3_hij=((blockSize-n_h)/(blockSize-1))*mean_1/mean_3
    numerator_hij=q_hij*numerator_term_1_hij+(1-q_hij)*(numerator_term_2_hij+
      numerator_term_3_hij)

    denominator_term_1_hij=1-p_missing_y_hi
    denominator_term_2_hij=(bloc_total_2-blockSize*denominator_term_1_hij)/(blockSize*(
      blockSize-1))
    denominator_term_3_hij=((blockSize-n_h)/(blockSize-1))*mean_2/mean_3
    denominator_hij=q_hij*denominator_term_1_hij+(1-q_hij)*(denominator_term_2_hij+
      denominator_term_3_hij)

    cond_mean_zhj=numerator_hij/denominator_hij

#-----#

```

```

# cond_mean_zhj_sq - sigmasq
# other_cond_mean
#-----#
other_numerator_term_1_hij=(1-p.missing_y_hi)*mu_hi^2
other_numerator_term_2_hij=(bloc_total_3-blockSize*other_numerator_term_1_hij)/(blockSize
*(blockSize-1))
other_numerator_term_3_hij=((blockSize-n_h)/(blockSize-1))*mean_4/mean_3
other_numerator_hij=q_hij*other_numerator_term_1_hij+(1-q_hij)*(other_numerator_term_2_hij
+other_numerator_term_3_hij)

other_cond_mean=other_numerator_hij/denominator_hij

#-----#
# Update the score
#-----#

totalScore=totalScore+(z_hj-cond_mean_zhj)^2-(other_cond_mean-cond_mean_zhj^2);
}
}
}

est_sigmaSq=totalScore/total_numPairs

return(est_sigmaSq);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
# 7: WLS PW with estimated variance
#-----#

PW1LinearScore2 = function(beta, est_beta, est_sigmaSq, pairs, numBlocks, blockSize, minCmp) {

totalScore=0;
beta_0=beta[1];
beta_1=beta[2];

est_beta_0=est_beta[1];
est_beta_1=est_beta[2];

pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
pairs_with_nomissing_x=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

all_mus=beta_0+beta_1*pairs[,1]
all_p.missing_y=pairs[,10]
all_p.missing_x=pairs[,9]

```

```

all_missing_x=pairs[,7]

mean_1=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)*all_mus/(1-
    all_p_missing_x))
mean_2=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)/(1-all_p_missing_x))
mean_3=mean((1-all_missing_x)*all_p_missing_x/(1-all_p_missing_x))

#-----#
est_all_mus=est_beta_0+est_beta_1*pairs[,1]
est_mean_1=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)*est_all_mus/(1-
    all_p_missing_x))
#est2_mean_1
est_mean_4=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)*(est_all_mus^2+
    est_sigmaSq)/(1-all_p_missing_x))

for(b in 1:numBlocks) {

    block_pairs=subset(pairs, pairs[,6]==b)
    n_h=blockSize-sum(subset(pairs, pairs[,6]==b)[,7])/blockSize

    block_pairs_nomissing=subset(pairs_with_nomissing, pairs_with_nomissing[,6]==b)
    block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing, block_pairs_nomissing[,4]>=
        minCmp);
    numPairs=nrow(block_pairs_nomissing_above_cmp);

    mus=beta_0+beta_1*block_pairs[,1]
    p_missing_y=block_pairs[,10]
    nonmissing_x=1-block_pairs[,7]

    bloc_total_1=sum(nonmissing_x*(1-p_missing_y)*mus)
    bloc_total_2=sum(nonmissing_x*(1-p_missing_y))

    #-----#
    est_mus=est_beta_0+est_beta_1*block_pairs[,1]
    est_bloc_total_1=sum(nonmissing_x*(1-p_missing_y)*est_mus)
    est2_bloc_total_1=sum(nonmissing_x*(1-p_missing_y)*(est_mus^2+est_sigmaSq))

    #-----#

    if(numPairs>0){

        for(r in 1:numPairs){

            #-----#
            # cond_mean_zhj
            #-----#

            mu_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
            p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
            z_hj=block_pairs_nomissing_above_cmp[r,2]
            q_hij=block_pairs_nomissing_above_cmp[r,4]

            numerator_E_z_term_1=(1-p_missing_y_hi)*mu_hi

```

```

    numerator_E_z_term_2=(bloc_total_1-blockSize*numerator_E_z_term_1)/(blockSize*(
        blockSize-1))
    numerator_E_z_term_3=((blockSize-n_h)/(blockSize-1))*mean_1/mean_3
    numerator_E_z=q_hij*numerator_E_z_term_1+(1-q_hij)*(numerator_E_z_term_2+
        numerator_E_z_term_3)

    denominator_E_z_term_1=1-p.missing_y_hi
    denominator_E_z_term_2=(bloc_total_2-blockSize*denominator_E_z_term_1)/(blockSize*(
        blockSize-1))
    denominator_E_z_term_3=((blockSize-n_h)/(blockSize-1))*mean_2/mean_3
    denominator_E_z=q_hij*denominator_E_z_term_1+(1-q_hij)*(denominator_E_z_term_2
        +denominator_E_z_term_3)

    cond_mean_zhj=numerator_E_z/denominator_E_z

#-----#
# est_cond.mean.zhj
#-----#
est_mu_hi=est_beta_0+est_beta_1*block_pairs_nomissing_above_cmp[r,1]

est_numerator_E_z_term_1=(1-p.missing_y_hi)*est_mu_hi
est_numerator_E_z_term_2=(est_bloc_total_1-blockSize*est_numerator_E_z_term_1)/(
    blockSize*(blockSize-1))
est_numerator_E_z_term_3=((blockSize-n_h)/(blockSize-1))*est_mean_1/mean_3
est_numerator_E_z=q_hij*est_numerator_E_z_term_1+(1-q_hij)*(
    est_numerator_E_z_term_2+est_numerator_E_z_term_3)

est_cond_mean_zhj=est_numerator_E_z/denominator_E_z

#-----#
# est_cond.mean.zhj_sq
#-----#
est_numerator_E_zsq_term_1=(1-p.missing_y_hi)*(est_mu_hi^2+est_sigmaSq)
est_numerator_E_zsq_term_2=(est2_bloc_total_1-blockSize*est_numerator_E_zsq_term_1)/(
    blockSize*(blockSize-1))
est_numerator_E_zsq_term_3=((blockSize-n_h)/(blockSize-1))*(est_mean_4)/mean_3
est_numerator_E_zsq=q_hij*est_numerator_E_zsq_term_1+(1-q_hij)*(
    est_numerator_E_zsq_term_2+est_numerator_E_zsq_term_3)

denominator_E_zsq=denominator_E_z
est_cond_mean_zhj_sq=est_numerator_E_zsq/denominator_E_zsq

#-----#
# Update the score
#-----#

totalScore=totalScore+(z_hj-cond_mean_zhj)^2/(est_cond_mean_zhj_sq-est_cond_mean_zhj
    ^2);
}
}
}

return(totalScore);

```

```

}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
#         9: PW2 least squares
#-----#

PW2LinearScore1 = function(beta, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
  pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

  all_mus=beta_0+beta_1*pairs[,1]
  all_p_missing_y=pairs[,10]
  all_p_missing_x=pairs[,9]
  all_missing_x=pairs[,7]

  mean_1=mean((1-all_missing_x)*(1-all_p_missing_y)*all_mus/(1-all_p_missing_x))
  mean_2=mean((1-all_missing_x)*(1-all_p_missing_y)/(1-all_p_missing_x))

  for(b in 1:numBlocks) {

    block_pairs=subset(pairs, pairs[,6]==b)

    block_pairs_nomissing=subset(pairs_with_nomissing, pairs_with_nomissing[,6]==b)
    block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing, block_pairs_nomissing[,4]>=
      minCmp);

    numPairs=nrow(block_pairs_nomissing_above_cmp);
    if (numPairs>0){
      for(r in 1:numPairs){
        mu_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
        p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
        z_hj=block_pairs_nomissing_above_cmp[r,2]
        q_hij=block_pairs_nomissing_above_cmp[r,4]

        numerator_E_z=q_hij*(1-p_missing_y_hi)*mu_hi+(1-q_hij)*mean_1
        denominator_E_z=q_hij*(1-p_missing_y_hi)+(1-q_hij)*mean_2
        cond_mean_zhj=numerator_E_z/denominator_E_z

        totalScore=totalScore+(z_hj-cond_mean_zhj)^2;
      }
    }
  }
}

```

```

    }

    return(totalScore);
}

#-----#
# Function: Estimate sigma-sq for PW linear
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
#
#-----#

PW2EstimateSigmasq = function(beta, pairs, numBlocks, blockSize, minCmp) {

    total_numPairs=0
    totalScore=0;
    beta_0=beta[1];
    beta_1=beta[2];

    pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
    pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

    all_mus=beta_0+beta_1*pairs[,1]
    all_p_missing_y=pairs[,10]
    all_p_missing_x=pairs[,9]
    all_missing_x=pairs[,7]

    mean_1=mean((1-all_missing_x)*(1-all_p_missing_y)*all_mus/(1-all_p_missing_x))
    mean_2=mean((1-all_missing_x)*(1-all_p_missing_y)/(1-all_p_missing_x))

    #est_mean_1=mean((1-all_missing_x)*(1-all_p_missing_y)*est_all_mus/(1-all_p_missing_x))
    #est2_mean_1
    mean_4=mean((1-all_missing_x)*(1-all_p_missing_y)*(all_mus^2)/(1-all_p_missing_x))

    for(b in 1:numBlocks) {

        block_pairs=subset(pairs, pairs[,6]==b)

        block_pairs_nomissing=subset(pairs_with_nomissing, pairs_with_nomissing[,6]==b)
        block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing, block_pairs_nomissing[,4]>=
            minCmp);

        numPairs=nrow(block_pairs_nomissing_above_cmp);

        total_numPairs=total_numPairs+numPairs

        if(numPairs>0){
            for(r in 1:numPairs){

#-----#

```

```

# est_cond_mean_zhj
#-----#
    mu_hi=beta.0+beta.1*block_pairs.nomissing.above_cmp[r,1]
    p_missing_y_hi=block_pairs.nomissing.above_cmp[r,10]
    z_hj=block_pairs.nomissing.above_cmp[r,2]
    q_hij=block_pairs.nomissing.above_cmp[r,4]

    numerator_E_z=q_hij*(1-p_missing_y_hi)*mu_hi+(1-q_hij)*mean.1
    denominator_E_z=q_hij*(1-p_missing_y_hi)+(1-q_hij)*mean.2
    cond_mean_zhj=numerator_E_z/denominator_E_z

#-----#
# est_cond_mean_zhj_sq
#-----#
    numerator_other=q_hij*(1-p_missing_y_hi)*(mu_hi^2)+(1-q_hij)*mean.4
    other_cond_mean=numerator_other/denominator_E_z

#-----#
# update score
#-----#
    totalScore=totalScore+(z_hj-cond_mean_zhj)^2-(other_cond_mean-cond_mean_zhj^2);
}
}
}

est_sigmaSq=totalScore/total_numPairs

return(est_sigmaSq);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
# 10: PW2 WLS with estimated variance
#-----#

PW2LinearScore2 = function(beta, est_beta, est_sigmaSq, pairs, numBlocks, blockSize, minCmp) {

    totalScore=0;
    beta_0=beta[1];
    beta_1=beta[2];

    est_beta_0=est_beta[1];
    est_beta_1=est_beta[2];

    pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
    pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

```

```

    all_mus=beta_0+beta_1*pairs[,1]
all_p_missing_y=pairs[,10]
all_p_missing_x=pairs[,9]
    all_missing_x=pairs[,7]

mean_1=mean((1-all_missing_x)*(1-all_p_missing_y)*all_mus/(1-all_p_missing_x))
mean_2=mean((1-all_missing_x)*(1-all_p_missing_y)/(1-all_p_missing_x))

est_all_mus=est_beta_0+est_beta_1*pairs[,1]
est_mean_1=mean((1-all_missing_x)*(1-all_p_missing_y)*est_all_mus/(1-all_p_missing_x))
#est2_mean_1
est_mean_4=mean((1-all_missing_x)*(1-all_p_missing_y)*(est_all_mus^2+est_sigmaSq)/(1-
    all_p_missing_x))

for(b in 1:numBlocks) {

    block_pairs=subset(pairs , pairs[,6]==b)
    block_pairs_nomissing=subset(pairs_with_nomissing , pairs_with_nomissing[,6]==b)
    block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing , block_pairs_nomissing[,4]>=
        minCmp);

    numPairs=nrow(block_pairs_nomissing_above_cmp);

    if (numPairs>0){
        for(r in 1:numPairs){

            #-----#
            # cond_mean_zhj
            #-----#
            mu_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
            p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
            z_hj=block_pairs_nomissing_above_cmp[r,2]
            q_hij=block_pairs_nomissing_above_cmp[r,4]

            numerator_E_z=q_hij*(1-p_missing_y_hi)*mu_hi+(1-q_hij)*mean_1
            denominator_E_z=q_hij*(1-p_missing_y_hi)+(1-q_hij)*mean_2
            cond_mean_zhj=numerator_E_z/denominator_E_z

            #-----#
            # est_cond_mean_zhj
            #-----#
            est_mu_hi=est_beta_0+est_beta_1*block_pairs_nomissing_above_cmp[r,1]
            est_numerator_E_z=q_hij*(1-p_missing_y_hi)*est_mu_hi+(1-q_hij)*est_mean_1
            est_cond_mean_zhj=est_numerator_E_z/denominator_E_z

            #-----#
            # est_cond_mean_zhj_sq
            #-----#
            est_numerator_E_zsq=q_hij*(1-p_missing_y_hi)*(est_mu_hi^2+est_sigmaSq)+(1-q_hij)*
                est_mean_4
            est_cond_mean_zhj_sq=est_numerator_E_zsq/denominator_E_z

            #-----#

```



```

    # update score
    #-----#
    totalScore=totalScore+(z_hj-cond_mean_zhj)^2/(est_cond_mean_zhj-sq-est_cond_mean_zhj^2);
  }
}
}

return(totalScore);
}

#-----#
# Function: minimize score
# pairs: matrix with covariate (x), observed response (z)
#       match status (m), cmp (q) and link status (l)
# scoreType:
#
#       0: complete data for linear and homoschedastic
#       1: naive BLUE for linear and homoschedastic
#
#       2: complete data for logistic
#       3: naive QL for logistic
#
#       4: complete data for survival
#       5: naive QL for survival
#
# 6: PW LSE linear homoschedastic
# 7: PW WLS linear homoschedastic
#-----#

minimizeScore = function(initBeta=c(0,0), est_beta=c(0,0), est_sigmaSq=1.0, pairs, followupTime=0,
  numBlocks, blockSize, minCmp=0.0, scoreOption) {

  if(scoreOption==0 | scoreOption==1) result <- optim(initBeta, linearScore, gr=NULL, pairs=pairs,
    scoreOption=scoreOption, minCmp=minCmp, method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==2) result <- optim(initBeta, PW1LinearScore1, gr=NULL, pairs=pairs,
    numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(
    fnscale=1))
  else if(scoreOption==3) result <- optim(initBeta, PW1LinearScore2, gr=NULL, est_beta=est_beta,
    est_sigmaSq=est_sigmaSq, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp
    , method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==4) result <- optim(initBeta, PW2LinearScore1, gr=NULL, pairs=pairs,
    numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(
    fnscale=1))
  else if(scoreOption==5) result <- optim(initBeta, PW2LinearScore2, gr=NULL, est_beta=est_beta,
    est_sigmaSq=est_sigmaSq, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp
    , method=c("BFGS"), control=list(fnscale=1))
  return(result);
}

#-----#
# Simulation parameters
#-----#

```

```

        numBlocks = 128;
        numRepetitions = 100;
        numIter = 10;
        beta = matrix(c(0.5,1.0),2,1);

    scenarioList = list();
    ScenarioResults = list()

scenarioList[[1]] = list(blockSize=8,numLinkVars=8);
scenarioList[[2]] = list(blockSize=4,numLinkVars=8);
scenarioList[[3]] = list(blockSize=8,numLinkVars=8);

sink("output.txt");

#-----#
# Actual simulations
#-----#

#-----#
# All estimates
# for each row
# -scenario
# -iteration
# -cmp
# -method: 1-14
# -estimated beta_0
# -estimated beta_1
#-----#

for(scen in 1:1) {

    blockSize = scenarioList[[scen]]$blockSize;
    numLinkVars = scenarioList[[scen]]$numLinkVars;

    allGammas = t(generateAllGammas(numLinkVars));

    for(t in 1:numRepetitions) {

        if(t%%5==0) {
            print(list(Iteration=t));
        }

        FPData = generateFinitePopulation(numBlocks=numBlocks, blockSize=blockSize, numLinkVars=
            numLinkVars);

        sigmasq=(FPData$sigma)^2;

        pairsData = generatePairs(FPData=FPData);
        popSize = FPData$popSize;
        blockSize = FPData$blockSize;
        numBlocks = FPData$numBlocks;

```

```

beta_missing_y_0=FPData$beta_missing_y_0;
beta_missing_y_1=FPData$beta_missing_y_1;

# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
#         block (b), (x is missing), (y is missing?),
#         p-missing-x, p-missing-y

potentialPairs = pairsData$potentialPairs;

#print(list(ncol=ncol(potentialPairs)))

pairs = matrix(rep(0,10*numBlocks*blockSize^2),numBlocks*blockSize^2,10);

# x
pairs[,1] = potentialPairs[(4+numLinkVars)];

# z
pairs[,2] = potentialPairs[(8+numLinkVars)];

# match status
pairs[,3] = potentialPairs[(15+numLinkVars)];

# cmp
pairs[,4] = potentialPairs[(13+numLinkVars)];

# link status
pairs[,5] = potentialPairs[(16+numLinkVars)];

# block
pairs[,6] = potentialPairs[,1];

# x is missing
pairs[,7] = potentialPairs[(5+numLinkVars)];

# y is missing
pairs[,8] = potentialPairs[(9+numLinkVars)];

# p-missing-x
pairs[,9] = potentialPairs[(6+numLinkVars)];

# p-missing-y
pairs[,10] = potentialPairs[(7+numLinkVars)];

#print(list(pairs=pairs[1:5,]))

initBeta=beta+c(rnorm(2,0,0.5));

minCmp = 0.9;

#-----#
# Naive estimator

```

```

# method 1
#-----#
scoreOption = 1;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp, scoreOption=scoreOption);

if(scen==1 && t==1) all_estimates=c(scen , t , 1 , result$par)
else all_estimates=rbind(all_estimates , c(scen , t , 1 , result$par))

#-----#
# Complete data
# method 2
#-----#
scoreOption = 0;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp, scoreOption=scoreOption);
all_estimates=rbind(all_estimates , c(scen , t , 2 , result$par))

#-----#
# PW1 LSE
# method 3
#-----#
scoreOption = 2;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
    minCmp=minCmp, scoreOption=scoreOption);
est_beta =result$par;

all_estimates=rbind(all_estimates , c(scen , t , 3 , result$par))

#-----#
# PW1 WLSE
# method 4
#-----#
est_sigmaSq=PW1EstimateSigmasq(est_beta , pairs , numBlocks , blockSize , minCmp)
scoreOption = 3;
result = minimizeScore(initBeta=initBeta , est_beta=est_beta , est_sigmaSq=est_sigmaSq , pairs=pairs
    , numBlocks=numBlocks , blockSize=blockSize , minCmp=minCmp, scoreOption=scoreOption);
all_estimates=rbind(all_estimates , c(scen , t , 4 , result$par))

#-----#
# PW2 LSE
# method 5
#-----#
scoreOption = 4;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize
    , minCmp=minCmp, scoreOption=scoreOption);
all_estimates=rbind(all_estimates , c(scen , t , 5 , result$par))
est_beta =result$par;

#-----#
# PW2 WLSE
# method 6
#-----#

```

```

    est_sigmaSq=PW2EstimateSigmasq(est_beta , pairs , numBlocks , blockSize , minCmp)
    scoreOption = 5;
    result = minimizeScore(initBeta=initBeta , est_beta=est_beta , est_sigmaSq=sigmasq , pairs=pairs ,
        numBlocks=numBlocks , blockSize=blockSize , minCmp=minCmp , scoreOption=scoreOption);
    all_estimates=rbind(all_estimates , c(scen , t , 6 , result$par))
  }

}

write.csv(all_estimates , file = "linear_all_estimates_missing_k8_Nh8_cmp90.csv")

numScen=1
allResults=perfAll(beta[1] , beta[2] , all_estimates , numScen)

write.csv(allResults , file = "linearResults_missing_k8_Nh8_cmp90.csv")

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

sink();

```

B.2.2 Logistic regression

The following R code was used.

```

#-----#
#
# randomPermutation(blockSize=)
#
#-----#

randomPermutation = function (blockSize)
{
    u = runif(blockSize,0,1);
    sortedU = sort(u);

    permutationMatrix = matrix(rep(0,blockSize^2),blockSize , blockSize);

    for (j in 1:blockSize)
        for (i in 1:blockSize) permutationMatrix[i,j] = (u[i]==sortedU[j]);

    return(permutationMatrix);
}

#-----#
#
# generateFinitePopulation(numBlocks=, blockSize=, numLinkVars=)
#

```

```

#-----#

generateFinitePopulation = function (numBlocks,
                                     blockSize,
                                     numLinkVars) # all binary variables
{

  ALPHA = 0.5;
  BETA = 1.0;
  beta_missing_y_0=-2.0;
  beta_missing_y_1=1.0;
  beta_missing_x_0=-2.0;
  beta_missing_x_1=1.0;
  # MEAN_X = 0.0;
  # SIGMA_X = 1.0;
  SIGMA = 0.7;
  P = 0.5;
  Q0 = 0.05;
  Q1 = 0.95;
  SHUFFLEPROBA = 1.0;

  #for low quality Q0=0.2, Q1=0.8
  #for medium quality Q0=0.1, Q1=0.9
  #for high quality Q0=0.05, Q1=0.95

  popSize = numBlocks*blockSize;

  blocks = c(t(matrix(rep(c(1:numBlocks),blockSize),numBlocks,blockSize)));

  mixingProportion = 1/blockSize;

  uAgree = (P*Q1+(1-P)*Q0)^2+(1-(P*Q1+(1-P)*Q0))^2;
  mAgree = P*(Q1^2+(1-Q1)^2)+(1-P)*((1-Q0)^2+Q0^2);

  #uAgree = 1/4;
  #mAgree = 1/2;

  x = runif(popSize,min=-1,max=1);
  eta = ALPHA+BETA*x
  mu = exp(eta)/(1+exp(eta))
  y = rbinom(popSize,1,mu)

  eta_missing_x=beta_missing_x_0+beta_missing_x_1*x;

  eta_missing_y=beta_missing_y_0+beta_missing_y_1*x;

  p_missing_x=exp(eta_missing_x)/(1+exp(eta_missing_x));

  p_missing_y=exp(eta_missing_y)/(1+exp(eta_missing_y));

  missing_x=rbinom(popSize,1,p_missing_x);
  missing_y=rbinom(popSize,1,p_missing_y);

```

```

origLinkVars = matrix(rbinom(popSize*numLinkVars,1,P),popSize,numLinkVars);

recordedLinkVarsA = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars,1,Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars,1,Q1),popSize,numLinkVars);
recordedLinkVarsB = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars,1,Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars,1,Q1),popSize,numLinkVars);

blockids = c(t(matrix(rep(c(1:numBlocks),blockSize), numBlocks, blockSize)));

recidA = cbind(blockids, matrix(rep(matrix(c(1:blockSize), blockSize, 1),numBlocks),popSize,1));
recidB = recidA;

#-----#
# Apply a random permutation to B records
#-----#

# Shuffle the linkage variables and the responses
# within each block
oRecidB = recidB;

matchMatrices = list();

for(b in 1:numBlocks) {

  startIndex = (b-1)*blockSize+1;
  endIndex = b*blockSize;
  permMat = diag(rep(1,blockSize));

  shuffleYes = rbinom(1,1,SHUFFLEPROBA);

  if(shuffleYes) {
                                permMat = randomPermutation(blockSize);
    oRecidB[startIndex:endIndex,1:2] = permMat%%recidB[startIndex:endIndex
,1:2];
    recordedLinkVarsB[startIndex:endIndex,1:numLinkVars] = permMat%%recordedLinkVarsB[startIndex:
endIndex,1:numLinkVars];
                                y[startIndex:endIndex] = permMat%%y[startIndex:endIndex];
                                missing_y[startIndex:endIndex] = permMat%%missing_y[startIndex:endIndex
];
  }

  matchMatrices[[b]] = permMat;

}

FPData = list(
  numBlocks = numBlocks,
  blockSize = blockSize,
  popSize = popSize,
  numLinkVars = numLinkVars,

```

```

        blocks = blocks ,
        recidA = recidA ,
        oRecidB = oRecidB ,
        recidB = recidB ,
        matchMatrices = matchMatrices ,
    origLinkVars      = origLinkVars ,
    recordedLinkVarsA = recordedLinkVarsA ,
    recordedLinkVarsB = recordedLinkVarsB ,
        shuffleProba = SHUFFLEPROBA,
        p  = P,
        q0 = Q0,
        q1 = Q1,
        mixingProportion = mixingProportion ,
    mAgree = mAgree ,
    uAgree = uAgree ,
        x  = x ,
        y  = y ,
    missing_x = missing_x ,
    missing_y = missing_y ,
    p.missing_x=p.missing_x ,
    p.missing_y=p.missing_y ,
        model = 'logistic ' ,
        alpha = ALPHA,
        beta  = BETA);

    return (FPData);
}

#-----#
#
# compareLinkVars(v1, v2)
#
#-----#

compareLinkVars = function(v1, v2) {

    numLinkVars = dim(v1)[2];
    numPairs = dim(v1)[1]

    gammas = matrix(rep(0,numLinkVars*numPairs),numPairs,numLinkVars);

    for(i in 1:numPairs)
        for(j in 1:numLinkVars) gammas[i,j] = (v1[i,j]==v2[i,j]);

    return(gammas);
}

#-----#
#
# generatePairs(FPData)
#
#-----#

```



```

generatePairs = function(FPData)
{

  popSize      = FPData$popSize;
  numBlocks    = FPData$numBlocks;
  blockSize    = FPData$blockSize;
  numLinkVars  = FPData$numLinkVars;
  recidA       = FPData$recidA;
  recidB       = FPData$recidB;
  oRecidB      = FPData$oRecidB;
  target_fnr   = 0.05;
  #missing_y   = FPData$missing_y;

  mixingProportion = FPData$mixingProportion;

  mAgree = FPData$mAgree;
  uAgree = FPData$uAgree;

  shuffleProba = FPData$shuffleProba;

  ncols      = 2+numLinkVars;
  ncols_A    = 5+numLinkVars;
  ncols_B    = 3+numLinkVars;

  #-----#
  # tableA: with ncols columns
  #       col 1: (perfect) blocking variable
  #       col 2 through col ncols-1: linkage variables
  #       col ncols: covariate x
  #
  # tableB: with ncols columns
  #       col 1: (perfect) blocking variable
  #       col 2 through col ncols-1: linkage variables
  #       col ncols: covariate y
  #-----#

  tableA = cbind(FPData$blocks, FPData$recordedLinkVarsA, FPData$x, FPData$missing_x,
    FPData$p_missing_x, FPData$p_missing_y);
  tableB = cbind(FPData$blocks, FPData$recordedLinkVarsB, FPData$y, FPData$missing_y);

  for(b in 1:numBlocks) {

    # Get the records in the block b in each file

    blockA=matrix(tableA[(tableA[,1]==b)], blockSize, ncols_A);
    blockB=matrix(tableB[(tableB[,1]==b)], blockSize, ncols_B);

    startIndex_b=(b-1)*blockSize+1;
    endIndex_b=b*blockSize;

    oRecidB_b=matrix(rep(0,blockSize),blockSize,1);
    oRecidB_b=oRecidB[startIndex_b:endIndex_b,2];
  }
}

```

```

for(r in 1:blockSize) {

#-----#
# For t the block size build
#
#      x_r y_1
#      x_r y_2
#      .   .
#      .   .
#      .   .
#      x_r y_t
#-----#

analyticalVars = cbind(t(matrix(rep(blockA[r,(ncols_A-3):ncols_A],blockSize),4,blockSize)),
      blockB[, (ncols_B-1):ncols_B]);

#-----#
# For t the block size build
# v_j the vector of linkage
#      variables for record j
#
#      gamma(v_r,v_1)
#      gamma(v_r,v_2)
#      .   .
#      .   .
#      .   .
#      gamma(v_r,v_t)
#-----#

gammas = compareLinkVars(t(matrix(rep(blockA[r,2:(ncols-1)],blockSize),numLinkVars, blockSize)
), blockB[,2:(ncols-1)]);
tmpMat = cbind(rep(b,blockSize), rep(r,blockSize), c(1:blockSize), gammas, analyticalVars,
      matrix(rep(0,5*blockSize),blockSize,5), 1*(oRecidB_b==r));

if(b==1 && r==1) potentialPairs_0=tmpMat else potentialPairs_0=rbind(potentialPairs_0, tmpMat)
;
}

}

#print(list(potentialPairs_0=potentialPairs_0))

#-----#
# E-M algorithm
#-----#

numPairs = numBlocks*blockSize^2;
pairsGammas = potentialPairs_0[,4:(4+numLinkVars-1)];
estParams = EMAlgorithm(numLinkVars=numLinkVars, blockSize=blockSize, pairsGammas=pairsGammas);

```

```

    lambda = estParams$lambda;
    m_probas = estParams$m_probas;
    u_probas = estParams$u_probas;

    m_gamma = rep(1,numPairs);
    u_gamma = rep(1,numPairs);

    for(k in 1:numLinkVars) {
        m_gamma = m_gamma*(m_probas[k]^pairsGammas[,k])*((1-m_probas[k])^(1-pairsGammas[,k]));
        u_gamma = u_gamma*(u_probas[k]^pairsGammas[,k])*((1-u_probas[k])^(1-pairsGammas[,k]));
    }

    w_gamma = log(m_gamma/u_gamma);
    q_gamma = lambda*m_gamma/(lambda*m_gamma+(1-lambda)*u_gamma);

    potentialPairs_0[,10+numLinkVars] = m_gamma;
    potentialPairs_0[,11+numLinkVars] = u_gamma;
    potentialPairs_0[,12+numLinkVars] = w_gamma;
    potentialPairs_0[,13+numLinkVars] = q_gamma;
    potentialPairs_0[,14+numLinkVars] = lambda;

    result=determineThreshold(numLinkVars, estParams, target_fnr)

    threshold=result$threshold

    potentialPairs=cbind(potentialPairs_0, (potentialPairs_0[, (12+numLinkVars)]>=threshold))

    #print(list(ncol=ncol(potentialPairs)))

    #-----#
    # Greedy linkage
    #-----#

    linkMatrices = list();

    for(b in 1:numBlocks) {

        # Get the records in the block b in each file

        blockA=matrix(tableA[(tableA[,1]==b)], blockSize, ncols_A);
        blockB=matrix(tableB[(tableB[,1]==b)], blockSize, ncols_B);

        selectionSofar = c();

        for(r in 1:blockSize) {

            startIndex = (b-1)*blockSize^2+(r-1)*blockSize+1;
            endIndex = (b-1)*blockSize^2+r*blockSize;
            w_gamma = potentialPairs[startIndex:endIndex,8+numLinkVars];

            tmpMat0 = cbind(matrix(c(1:blockSize), blockSize, 1), c(w_gamma));
            tmpMat1 = matrix(tmpMat0[!(tmpMat0[,1] %in% selectionSofar),], ncol=2);

```

```

        max_w = max(tmpMat1[,2]);
        candidates = matrix(tmpMat1[(tmpMat1[,2]==max_w)], ncol=2);
        num_candidates = dim(candidates)[1];

for(t in 1:num_candidates) {

    q = 1/(num_candidates-t+1);
    draw = rbinom(1,1,q);

    if(draw==1) {
        linkedRecidB = candidates[t,1];
        break;
    }

}

selectionSofar = c(selectionSofar, linkedRecidB);

if(r==1) linkMatrix = matrix((c(1:blockSize)==linkedRecidB)*1, blockSize, 1)
else linkMatrix = cbind(linkMatrix, matrix((c(1:blockSize)==linkedRecidB)*1, blockSize, 1));

}

linkMatrices[[b]] = linkMatrix;

}

#-----#
# Final output
#-----#

pairsData = list(
    numLinkVars = numLinkVars,
    mixingProportion = FPDData$mixingProportion,
    lambda = lambda,
    m_probas = m_probas,
    u_probas = u_probas,
    linkMatrices = linkMatrices,
    potentialPairs = potentialPairs);

return(pairsData);

}

#-----#
# all gammas: a recursive function
#-----#

generateAllGammas = function(k)
{
    if(k==1) return(c(0,1))
    else {
        prevGammas = generateAllGammas(k-1);

```

```

    allGammas = rbind(matrix(rep(prevGammas, 2), k-1, 2^k), c(rep(0, 2^(k-1)), rep(1, 2^(k-1))));
    return(allGammas);
  }
}

#-----#
# E-M algorithm
#-----#

EMAlgorithm = function (numLinkVars, blockSize, pairsGammas)
{
  lambda = 1/blockSize;
  initMproba = runif(1, min=0.75, max=1.0);
  initUproba = runif(1, min=0.0, max=0.25);

  m_probas = matrix(rep(initMproba, numLinkVars), 1, numLinkVars);
  u_probas = matrix(rep(initUproba, numLinkVars), 1, numLinkVars);

  dFrame = as.data.frame(pairsGammas);
  freqTable1 = as.data.frame(ftable(dFrame));
  freqTable2 = freqTable1[(freqTable1$Freq>0),];

  numProfiles=nrow(freqTable2);

  for(col in 1:numLinkVars) {

    if(col==1) {
      profilesFreqs = matrix(c(freqTable2[,col])-1, numProfiles, 1);
    }
    else {
      profilesFreqs = cbind(profilesFreqs, c(freqTable2[,col])-1);
    }
  }

  profilesFreqs = cbind(profilesFreqs, c(freqTable2$Freq), rep(1, numProfiles), rep(1, numProfiles),
    rep(0, numProfiles));

  numIter = 100;

  for(iter in 1:numIter) {

    estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

    profilesFreqs[,2+numLinkVars] = rep(1, numProfiles);
    profilesFreqs[,3+numLinkVars] = rep(1, numProfiles);

    for(k in 1:numLinkVars) {
      profilesFreqs[,2+numLinkVars] = profilesFreqs[,2+numLinkVars]*(m_probas[k]^profilesFreqs[,k])
        *((1-m_probas[k])^(1-profilesFreqs[,k]));
      profilesFreqs[,3+numLinkVars] = profilesFreqs[,3+numLinkVars]*(u_probas[k]^profilesFreqs[,k])
        *((1-u_probas[k])^(1-profilesFreqs[,k]));
    }
  }
}

```

```

profilesFreqs[,4+numLinkVars] = lambda*profilesFreqs[,2+numLinkVars]/(lambda*profilesFreqs[,2+
numLinkVars]+(1-lambda)*profilesFreqs[,3+numLinkVars]);

for(k in 1:numLinkVars) {
  m_probas[k] =sum(profilesFreqs[,k]*profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars])
    /sum(profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars]);
  u_probas[k] =sum(profilesFreqs[,k]*(1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+
numLinkVars])/sum((1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+numLinkVars]);
}

}

estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

return(estParams);

}

#-----#
# determineThreshold:
#-----#

determineThreshold = function (numLinkVars, rlParams, target_fnr)
{
  allGammas<-generateAllGammas(numLinkVars)

  numProfiles<-2^numLinkVars

  lambda = rlParams$lambda;
  m_probas = rlParams$m_probas;
  u_probas = rlParams$u_probas;

  m_gamma = rep(1,numProfiles);
  u_gamma = rep(1,numProfiles);

  for(k in 1:numLinkVars) {
    m_gamma = m_gamma*(m_probas[k]^allGammas[k,])*((1-m_probas[k])^(1-allGammas[k,]));
    u_gamma = u_gamma*(u_probas[k]^allGammas[k,])*((1-u_probas[k])^(1-allGammas[k,]));
  }

  w_gamma = log(m_gamma/u_gamma);

  weight_order<-order(w_gamma);
  sum_m=m_gamma[weight_order[1]];

  t=1;
  while(sum_m<=target_fnr && t<numProfiles){
    t=t+1
    sum_m=sum_m+m_gamma[weight_order[t]]
  }
  if(t==1) threshold=w_gamma[weight_order[1]]
  else if(sum_m>target_fnr && t>1) threshold=w_gamma[weight_order[t-1]]
}

```

```

else threshold=w_gamma[weight_order[t]]

est_fnr=sum(m_gamma*(w_gamma<threshold))
est_fpr=sum(u_gamma*(w_gamma>=threshold))

result=list(threshold=threshold, est_fnr=est_fnr, est_fpr=est_fpr)

return(result)
}

#-----#
# Performance measures
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfOne = function(beta_0, beta_1, all_estimates, scen, estMethod) {

  selectedscen=subset(all_estimates, (all_estimates[,1]==scen));
  selectedEstimates=subset(selectedscen, (selectedscen[,3]==estMethod));

  numRows=nrow(selectedEstimates);
  bias_beta0=round(100*mean(selectedEstimates[,4] - beta_0)/beta_0,3);
  mse_beta0=round(mean((selectedEstimates[,4] - beta_0)^2),6);
  var_beta0=round((sum((selectedEstimates[,4] - mean(selectedEstimates[,4]))^2)/(numRows-1)),6);

  bias_beta1=round(100*mean(selectedEstimates[,5] - beta_1)/beta_1,3);
  mse_beta1=round(mean((selectedEstimates[,5] - beta_1)^2),6);
  var_beta1=round((sum((selectedEstimates[,5] - mean(selectedEstimates[,5]))^2)/(numRows-1)),6);

  result=rbind(c(scen,estMethod,bias_beta0,var_beta0,mse_beta0),c(scen,estMethod,bias_beta1,
    var_beta1,mse_beta1));
  return(result)
}

#-----#
# Performance measure for all methods
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfAll = function(beta_0, beta_1, all_estimates, numScen) {

  for(scen in 1:numScen) {
    if(scen==1) allResults=perfOne(beta_0, beta_1, all_estimates,scen,1)
    else allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,1));
    #allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,3))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,4))
    #allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,5))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,6))
    allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,2))
  }
}

```

```

    for(t in 1:4) {
      if(t==1) finalResults=allResults[1,]
      else finalResults=rbind(finalResults, allResults[2*(t-1)+1,])
    }
    for(t in 1:4) finalResults=rbind(finalResults, allResults[2*t,]);

    return(finalResults)
  }

#-----#
# Function: logit score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
# scoreOption:
#     0: complete data for linear and homoschedastic
#     1: naive BLUE for linear and homoschedastic
# y is missing at random. Ignore the observations where y
# is missing
#-----#

logitScore = function(beta, pairs, minCmp, scoreOption) {

  total=c(0,0);
  beta_0=beta[1];
  beta_1=beta[2];

  pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
  pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

  numPairs=nrow(pairs_with_nomissing);

  for(r in 1:numPairs){
    x_hi=pairs_with_nomissing[r,1];
    z_hj=pairs_with_nomissing[r,2];
    m_hij=pairs_with_nomissing[r,3];
    l_hij=(pairs_with_nomissing[r,4]>=minCmp);
    eta_hi=beta_0+beta_1*x_hi;
    mu_hi=exp(eta_hi)/(1+exp(eta_hi));
    total=total+((scoreOption==0)*m_hij+(scoreOption==1)*l_hij)*(z_hj-mu_hi)*c(1,x_hi);
  }

  score=sum(total*total)
  return(score);
}

#-----#
# Function: Pairwise logit score
#-----#

PWLogitScore1 = function(beta, pairs, numBlocks, blockSize, minCmp) {

```



```

totalScore=0;
beta_0=beta[1];
beta_1=beta[2];

pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

all_etas=beta_0+beta_1*pairs[,1]
all_mus=exp(all_etas)/(1+exp(all_etas))
all_p_missing_y=pairs[,10]
all_p_missing_x=pairs[,9]
all_missing_x=pairs[,7]

mean_1=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)*all_mus/(1-all_p_missing_x))
mean_2=mean((1-all_missing_x)*all_p_missing_x*(1-all_p_missing_y)/(1-all_p_missing_x))
mean_3=mean((1-all_missing_x)*all_p_missing_x/(1-all_p_missing_x))

for(b in 1:numBlocks) {

  block_pairs=subset(pairs, pairs[,6]==b)
  n_h=blockSize-sum(subset(pairs, pairs[,6]==b)[,7])/blockSize

  block_pairs_nomissing=subset(pairs_with_nomissing, pairs_with_nomissing[,6]==b)
  block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing, block_pairs_nomissing[,4]>=
    minCmp);

  numPairs=nrow(block_pairs_nomissing_above_cmp);

  etas=beta_0+beta_1*block_pairs[,1]
  mus=exp(etas)/(1+exp(etas))
  p_missing_y=block_pairs[,10]
  nonmissing_x=1-block_pairs[,7]

  bloc_total_1=sum(nonmissing_x*(1-p_missing_y)*mus)
  bloc_total_2=sum(nonmissing_x*(1-p_missing_y))

  if(numPairs>0){
    for(r in 1:numPairs){
      eta_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
      mu_hi=exp(eta_hi)/(1+exp(eta_hi))
      p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
      z_hj=block_pairs_nomissing_above_cmp[r,2]
      q_hij=block_pairs_nomissing_above_cmp[r,4]

      numerator_term_1_hij=(1-p_missing_y_hi)*mu_hi
      numerator_term_2_hij=(bloc_total_1-blockSize*numerator_term_1_hij)/(blockSize*(blockSize
        -1))
      numerator_term_3_hij=((blockSize-n_h)/(blockSize-1))*mean_1/mean_3
      numerator_hij=q_hij*numerator_term_1_hij+(1-q_hij)*(numerator_term_2_hij+
        numerator_term_3_hij)

      denominator_term_1_hij=1-p_missing_y_hi

```

```

denominator_term_2_hij=(bloc_total_2-blockSize*denominator_term_1_hij)/(blockSize*(
    blockSize-1))
denominator_term_3_hij=((blockSize-n_h)/(blockSize-1))*mean_2/mean_3
denominator_hij=q_hij*denominator_term_1_hij+(1-q_hij)*(denominator_term_2_hij+
    denominator_term_3_hij)

cond_mean_zhj=numerator_hij/denominator_hij

totalScore=totalScore+(z_hj-cond_mean_zhj)^2;
    }
}
}

return(totalScore);
}

#-----#
# Function: Pairwise logistic score
#-----#

PW1LogitScore2 = function(beta, est_beta, pairs, numBlocks, blockSize, minCmp) {

totalScore=0;
    beta_0=beta[1];
    beta_1=beta[2];

    est_beta_0=est_beta[1];
    est_beta_1=est_beta[2];

    pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
    pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

    all_etas=beta_0+beta_1*pairs[,1]
    all_mus=exp(all_etas)/(1+exp(all_etas))
    all_p.missing.y=pairs[,10]
    all_p.missing.x=pairs[,9]
    all.missing.x=pairs[,7]

    mean_1=mean((1-all.missing.x)*all_p.missing.x*(1-all_p.missing.y)*all_mus/(1-
        all_p.missing.x))
    mean_2=mean((1-all.missing.x)*all_p.missing.x*(1-all_p.missing.y)/(1-all_p.missing.x))
    mean_3=mean((1-all.missing.x)*all_p.missing.x/(1-all_p.missing.x))

    est_all_etas=est_beta_0+est_beta_1*pairs[,1]
    est_all_mus=exp(est_all_etas)/(1+exp(est_all_etas))
    est_mean_1=mean((1-all.missing.x)*all_p.missing.x*(1-all_p.missing.y)*est_all_mus/(1-
        all_p.missing.x))
    est_mean_4=mean((1-all.missing.x)*all_p.missing.x*(1-all_p.missing.y)*(est_all_mus^2+
        est_all_mus*(1-est_all_mus))/(1-all_p.missing.x))

    for(b in 1:numBlocks) {

        block_pairs=subset(pairs, pairs[,6]==b)

```

```

n_h=blockSize-sum(subset(pairs , pairs[,6]==b)[,7])/blockSize

block_pairs_nomissing=subset(pairs_with_nomissing , pairs_with_nomissing[,6]==b)
block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing , block_pairs_nomissing[,4]>=
minCmp);

numPairs=nrow(block_pairs_nomissing_above_cmp);

etas=beta_0+beta_1*block_pairs[,1]
mus=exp(etas)/(1+exp(etas))
p_missing_y=block_pairs[,10]
nonmissing_x=1-block_pairs[,7]

bloc_total_1=sum(nonmissing_x*(1-p_missing_y)*mus)
bloc_total_2=sum(nonmissing_x*(1-p_missing_y))

est_etas=est_beta_0+est_beta_1*block_pairs[,1]
est_mus=exp(est_etas)/(1+exp(est_etas))
est_bloc_total_1=sum(nonmissing_x*(1-p_missing_y)*est_mus)
#est2_bloc_total_1
est_bloc_total_3=sum(nonmissing_x*(1-p_missing_y)*(est_mus^2+est_mus*(1-est_mus)))

if (numPairs>0){

for (r in 1:numPairs){

#-----#
# cond.mean.zhj
#-----#

eta_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
mu_hi=exp(eta_hi)/(1+exp(eta_hi))
p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
z_hj=block_pairs_nomissing_above_cmp[r,2]
q_hij=block_pairs_nomissing_above_cmp[r,4]

numerator_E_z_term_1=(1-p_missing_y_hi)*mu_hi
numerator_E_z_term_2=(bloc_total_1-blockSize*numerator_E_z_term_1)/(blockSize*(
blockSize-1))
numerator_E_z_term_3=((blockSize-n_h)/(blockSize-1))*mean_1/mean_3
numerator_E_z=q_hij*numerator_E_z_term_1+(1-q_hij)*(numerator_E_z_term_2+
numerator_E_z_term_3)

denominator_E_z_term_1=1-p_missing_y_hi
denominator_E_z_term_2=(bloc_total_2-blockSize*denominator_E_z_term_1)/(blockSize*(
blockSize-1))
denominator_E_z_term_3=((blockSize-n_h)/(blockSize-1))*mean_2/mean_3
denominator_E_z=q_hij*denominator_E_z_term_1+(1-q_hij)*(denominator_E_z_term_2
+denominator_E_z_term_3)

cond.mean.zhj=numerator_E_z/denominator_E_z

#-----#
# est.cond.mean.zhj
#-----#

```

```

        est_eta_hi=est_beta_0+est_beta_1*block_pairs_nomissing_above_cmp[r,1]
        est_mu_hi=exp(est_eta_hi)/(1+exp(est_eta_hi))

    est_numerator_E_z_term_1=(1-p_missing_y_hi)*est_mu_hi
    est_numerator_E_z_term_2=(est_bloc_total_1-blockSize*est_numerator_E_z_term_1)/(
        blockSize*(blockSize-1))
    est_numerator_E_z_term_3=((blockSize-n_h)/(blockSize-1))*est_mean_1/mean_3
    est_numerator_E_z=q_hij*est_numerator_E_z_term_1+(1-q_hij)*(
        est_numerator_E_z_term_2+est_numerator_E_z_term_3)

    est_cond_mean_zhj=est_numerator_E_z/denominator_E_z

#-----#
# est_cond_mean_zhj_sq
#-----#
    est_numerator_E_zsq_term_1=(1-p_missing_y_hi)*(est_mu_hi^2+est_mu_hi*(1-est_mu_hi))
    est_numerator_E_zsq_term_2=(est_bloc_total_3-blockSize*est_numerator_E_zsq_term_1)/(
        blockSize*(blockSize-1))
    est_numerator_E_zsq_term_3=((blockSize-n_h)/(blockSize-1))*(est_mean_4)/mean_3
    est_numerator_E_zsq=q_hij*est_numerator_E_zsq_term_1+(1-q_hij)*(
        est_numerator_E_zsq_term_2+est_numerator_E_zsq_term_3)

    denominator_E_zsq=denominator_E_z
    est_cond_mean_zhj_sq=est_numerator_E_zsq/denominator_E_zsq

#-----#
# Update the score
#-----#

    totalScore=totalScore+(z_hj-cond_mean_zhj)^2/(est_cond_mean_zhj_sq-est_cond_mean_zhj
        ^2);
}
}
}

return(totalScore);
}

#-----#
# Function: Pairwise logistic score
#-----#

PW2LogitScore1 = function(beta, pairs, numBlocks, blockSize, minCmp) {

    totalScore=0;
    beta_0=beta[1];
    beta_1=beta[2];

    pairs_with_nomissing_x=subset(pairs, pairs[,7]==0)
    pairs_with_nomissing=subset(pairs_with_nomissing_x, pairs_with_nomissing_x[,8]==0)

    all_etas=beta_0+beta_1*pairs[,1]
    all_mus=exp(all_etas)/(1+exp(all_etas))

```

```

all_p_missing_y=pairs[,10]
all_p_missing_x=pairs[,9]
all_missing_x=pairs[,7]

mean_1=mean((1-all_missing_x)*(1-all_p_missing_y)*all_mus/(1-all_p_missing_x))
mean_2=mean((1-all_missing_x)*(1-all_p_missing_y)/(1-all_p_missing_x))

for(b in 1:numBlocks) {

  block_pairs=subset(pairs , pairs[,6]==b)

  block_pairs_nomissing=subset(pairs_with_nomissing , pairs_with_nomissing[,6]==b)
  block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing , block_pairs_nomissing[,4]>=
    minCmp);

  numPairs=nrow(block_pairs_nomissing_above_cmp);

  if (numPairs>0){
    for(r in 1:numPairs){
      eta_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
      mu_hi=exp(eta_hi)/(1+exp(eta_hi))
      p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
      z_hj=block_pairs_nomissing_above_cmp[r,2]
      q_hij=block_pairs_nomissing_above_cmp[r,4]

      numerator_E_z=q_hij*(1-p_missing_y_hi)*mu_hi+(1-q_hij)*mean_1
      denominator_E_z=q_hij*(1-p_missing_y_hi)+(1-q_hij)*mean_2
      cond_mean_zhj=numerator_E_z/denominator_E_z

      totalScore=totalScore+(z_hj-cond_mean_zhj)^2;
    }
  }
}

return(totalScore);
}

#-----#
# Function: Pairwise logistic score
#-----#

PW2LogitScore2 = function(beta , est_beta , pairs , numBlocks , blockSize , minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  est_beta_0=est_beta[1];
  est_beta_1=est_beta[2];

  pairs_with_nomissing_x=subset(pairs , pairs[,7]==0)
  pairs_with_nomissing=subset(pairs_with_nomissing_x , pairs_with_nomissing_x[,8]==0)

  all_etas=beta_0+beta_1*pairs[,1]
  all_mus=exp(all_etas)/(1+exp(all_etas))

```

```

all_p_missing_y=pairs[,10]
all_p_missing_x=pairs[,9]
all_missing_x=pairs[,7]

mean_1=mean((1-all_missing_x)*(1-all_p_missing_y)*all_mus/(1-all_p_missing_x))
mean_2=mean((1-all_missing_x)*(1-all_p_missing_y)/(1-all_p_missing_x))

est_all_etas=est_beta_0+est_beta_1*pairs[,1]
est_all_mus=exp(est_all_etas)/(1+exp(est_all_etas))
est_mean_1=mean((1-all_missing_x)*(1-all_p_missing_y)*est_all_mus/(1-all_p_missing_x))
#est2_mean_1
est_mean_4=mean((1-all_missing_x)*(1-all_p_missing_y)*(est_all_mus^2+est_all_mus*(1-est_all_mus))
                )/(1-all_p_missing_x))

for(b in 1:numBlocks) {

    block_pairs=subset(pairs, pairs[,6]==b)
    block_pairs_nomissing=subset(pairs-with-nomissing, pairs-with-nomissing[,6]==b)
    block_pairs_nomissing_above_cmp=subset(block_pairs_nomissing, block_pairs_nomissing[,4]>=
        minCmp);

    numPairs=nrow(block_pairs_nomissing_above_cmp);

    if (numPairs>0){
        for(r in 1:numPairs){

            #-----#
            # cond_mean_zhj
            #-----#
            eta_hi=beta_0+beta_1*block_pairs_nomissing_above_cmp[r,1]
            mu_hi=exp(eta_hi)/(1+exp(eta_hi))
            p_missing_y_hi=block_pairs_nomissing_above_cmp[r,10]
            z_hj=block_pairs_nomissing_above_cmp[r,2]
            q_hij=block_pairs_nomissing_above_cmp[r,4]

            numerator_E_z=q_hij*(1-p_missing_y_hi)*mu_hi+(1-q_hij)*mean_1
            denominator_E_z=q_hij*(1-p_missing_y_hi)+(1-q_hij)*mean_2
            cond_mean_zhj=numerator_E_z/denominator_E_z

            #-----#
            # est_cond_mean_zhj
            #-----#
            est_eta_hi=est_beta_0+est_beta_1*block_pairs_nomissing_above_cmp[r,1]
            est_mu_hi=exp(est_eta_hi)/(1+exp(est_eta_hi))
            est_numerator_E_z=q_hij*(1-p_missing_y_hi)*est_mu_hi+(1-q_hij)*est_mean_1
            est_cond_mean_zhj=est_numerator_E_z/denominator_E_z

            #-----#
            # est_cond_mean_zhj_sq
            #-----#
            est_numerator_E_zsq=q_hij*(1-p_missing_y_hi)*(est_mu_hi^2+est_mu_hi*(1-est_mu_hi))+(1-
                q_hij)*est_mean_4
            est_cond_mean_zhj_sq=est_numerator_E_zsq/denominator_E_z

```

```

#-----#
# update score
#-----#
totalScore=totalScore+(z-hj-cond.mean.zhj)^2/(est.cond.mean.zhj-sq-est.cond.mean.zhj^2);
}
}
}

return(totalScore);
}

#-----#
# Function: minimize score
# pairs: matrix with covariate (x), observed response (z)
# match status (m), cmp (q) and link status (l)
# scoreType:
#
#-----#

minimizeScore = function(initBeta=c(0,0), est.beta=c(0,0), pairs, followupTime=0, numBlocks,
  blockSize, minCmp=0.0, scoreOption) {

  if(scoreOption==0 | scoreOption==1) result <- optim(initBeta, logitScore, gr=NULL, pairs=pairs,
    scoreOption=scoreOption, minCmp=minCmp, method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==2) result <- optim(initBeta, PW1LogitScore1, gr=NULL, pairs=pairs, numBlocks
    =numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(fnscale=1))
  else if(scoreOption==3) result <- optim(initBeta, PW1LogitScore2, gr=NULL, est.beta=est.beta,
    pairs=pairs, numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"),
    control=list(fnscale=1))
  else if(scoreOption==4) result <- optim(initBeta, PW2LogitScore1, gr=NULL, pairs=pairs,
    numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"), control=list(
    fnscale=1))
  else if(scoreOption==5) result <- optim(initBeta, PW2LogitScore2, gr=NULL, est.beta=est.beta,
    pairs=pairs, numBlocks=numBlocks, blockSize=blockSize, minCmp=minCmp, method=c("BFGS"),
    control=list(fnscale=1))
  return(result);
}

#-----#
# Simulation parameters
#-----#

numBlocks = 128;
numRepetitions = 100;
numIter = 10;
beta = matrix(c(0.5,1.0),2,1);

scenarioList = list();
ScenarioResults = list()

scenarioList[[1]]=list(blockSize=8,numLinkVars=8);
scenarioList[[2]]=list(blockSize=4,numLinkVars=8);
scenarioList[[3]]=list(blockSize=8,numLinkVars=8);

```

```

sink("output.txt");

#-----#
# Actual simulations
#-----#

#-----#
# All estimates
# for each row
# -scenario
# -iteration
# -cmp
# -method: 1-14
# -estimated beta_0
# -estimated beta_1
#-----#

for(scen in 1:1) {

  blockSize = scenarioList[[scen]]$blockSize;
  numLinkVars = scenarioList[[scen]]$numLinkVars;

  allGammas = t(generateAllGammas(numLinkVars));

  for(t in 1:numRepetitions) {

    if(t%%5==0) {
      print(list(Iteration=t));
    }

    FPData = generateFinitePopulation(numBlocks=numBlocks, blockSize=blockSize, numLinkVars=
      numLinkVars);

    pairsData = generatePairs(FPData=FPData);
    popSize = FPData$popSize;
    blockSize = FPData$blockSize;
    numBlocks = FPData$numBlocks;

    beta_missing_y_0=FPData$beta_missing_y_0;
    beta_missing_y_1=FPData$beta_missing_y_1;

    # pairs: matrix with covariate (x), observed response (z)
    #       match status (m), cmp (q) and link status (l)
    #       block (b), (x is missing), (y is missing?),
    #       p-missing-x, p-missing-y

    potentialPairs = pairsData$potentialPairs;

    #print(list(ncol=ncol(potentialPairs)))
  }
}

```



```

pairs = matrix(rep(0,10*numBlocks*blockSize^2),numBlocks*blockSize^2,10);

# x
pairs[,1] = potentialPairs[(4+numLinkVars)];

# z
pairs[,2] = potentialPairs[(8+numLinkVars)];

# match status
pairs[,3] = potentialPairs[(15+numLinkVars)];

# cmp
pairs[,4] = potentialPairs[(13+numLinkVars)];

# link status
pairs[,5] = potentialPairs[(16+numLinkVars)];

# block
pairs[,6] = potentialPairs[,1];

# x is missing
pairs[,7] = potentialPairs[(5+numLinkVars)];

# y is missing
pairs[,8] = potentialPairs[(9+numLinkVars)];

# p-missing-x
pairs[,9] = potentialPairs[(6+numLinkVars)];

# p-missing-y
pairs[,10] = potentialPairs[(7+numLinkVars)];

#print(list(pairs=pairs[1:5,]))

initBeta=beta+c(rnorm(2,0,0.5));

minCmp = 0.9;

#-----#
# Naive estimator
# method 1
#-----#
scoreOption = 1;
result = minimizeScore(initBeta=initBeta,pairs=pairs,numBlocks=numBlocks,blockSize=blockSize,
minCmp=minCmp,scoreOption=scoreOption);

if(scen==1 && t==1) all_estimates=c(scen, t, 1, result$par)
else all_estimates=rbind(all_estimates,c(scen, t, 1, result$par))

#-----#
# Complete data
# method 2
#-----#

```

```

scoreOption = 0;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
  minCmp=minCmp , scoreOption=scoreOption);
all_estimates=rbind(all_estimates , c(scen , t , 2 , result$par))

#-----#
# PW1 LSE
# method 3
#-----#
scoreOption = 2;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize ,
  minCmp=minCmp , scoreOption=scoreOption);
est_beta =result$par;

all_estimates=rbind(all_estimates , c(scen , t , 3 , result$par))

#-----#
# PW1 WLSE
# method 4
#-----#
scoreOption = 3;
result = minimizeScore(initBeta=initBeta , est_beta=est_beta , pairs=pairs , numBlocks=numBlocks ,
  blockSize=blockSize , minCmp=minCmp , scoreOption=scoreOption);
all_estimates=rbind(all_estimates , c(scen , t , 4 , result$par))

#-----#
# PW2 LSE
# method 5
#-----#
scoreOption = 4;
result = minimizeScore(initBeta=initBeta , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize
  , minCmp=minCmp , scoreOption=scoreOption);
all_estimates=rbind(all_estimates , c(scen , t , 5 , result$par))
est_beta =result$par;

#-----#
# PW2 WLSE
# method 6
#-----#
scoreOption = 5;
result = minimizeScore(initBeta=initBeta , est_beta=est_beta , pairs=pairs , numBlocks=numBlocks ,
  blockSize=blockSize , minCmp=minCmp , scoreOption=scoreOption);
all_estimates=rbind(all_estimates , c(scen , t , 6 , result$par))
}

}

write.csv(all_estimates , file = "logit_all_estimates_missing_k8_Nh8_cmp90.csv")

numScen=1
allResults=perfAll(beta[1] , beta[2] , all_estimates , numScen)

write.csv(allResults , file = "logitResults_missing_k8_Nh8_cmp90.csv")

```

```
#-----#
#
#-----#

sink();
```

B.2.3 Survival model

The following R code was used.

```
#-----#
# randomPermutation(blockSize=)
#-----#

randomPermutation = function (blockSize)
{
    u = runif(blockSize,0,1);
    sortedU = sort(u);

    #i=perm(j)

    permutationMatrix = matrix(rep(0,blockSize^2),blockSize,blockSize);

    for (j in 1:blockSize)

        for (i in 1:blockSize) permutationMatrix[i,j] = (u[i]==sortedU[j]);

    return(permutationMatrix);
}

#-----#
# generateFinitePopulation(numBlocks=, blockSize=, numLinkVars=)
#-----#

generateFinitePopulation = function (numBlocks,
                                     blockSize,
                                     numLinkVars) # all binary variables
{

    ALPHA = 0.5;
    BETA = 1.0;
    # beta.missing-y.0=-2.0;
    # beta.missing-y.1=1.0;
    # beta.missing-x.0=-2.0;
    # beta.missing-x.1=1.0;
    P = 0.5;
    Q0 = 0.05;
```

```

Q1      = 0.95;
SHUFFLEPROBA = 1.0;
followupTime = 2.0;

#for low quality Q0=0.2, Q1=0.8
#for medium quality Q0=0.1, Q1=0.9
#for high quality Q0=0.05, Q1=0.95

popSize = numBlocks*blockSize;

blocks = c(t(matrix(rep(c(1:numBlocks), blockSize), numBlocks, blockSize)));

mixingProportion = 1/blockSize;

uAgree = (P*Q1+(1-P)*Q0)^2+(1-(P*Q1+(1-P)*Q0))^2;
mAgree = P*(Q1^2+(1-Q1)^2)+(1-P)*((1-Q0)^2+Q0^2);

#uAgree = 1/4;
#mAgree = 1/2;

x = 2*rbinom(popSize, 1, 0.5);
eta = ALPHA+BETA*x;
survivalTimes = -exp(-eta)*log(1-runif(popSize, 0, 1));
censored = (survivalTimes>followupTime)
y = followupTime*censored+(1-censored)*survivalTimes

# eta_missing_x=beta_missing_x_0+beta_missing_x_1*x;
#
# eta_missing_y=beta_missing_y_0+beta_missing_y_1*x;
#
# p_missing_x=exp(eta_missing_x)/(1+exp(eta_missing_x));
#
# p_missing_y=exp(eta_missing_y)/(1+exp(eta_missing_y));
#
# missing_x=rbinom(popSize, 1, p_missing_x);
# missing_y=rbinom(popSize, 1, p_missing_y);

origLinkVars = matrix(rbinom(popSize*numLinkVars, 1, P), popSize, numLinkVars);

recordedLinkVarsA = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars, 1, Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars, 1, Q1), popSize, numLinkVars);
recordedLinkVarsB = matrix((c(origLinkVars)==0)*rbinom(popSize*numLinkVars, 1, Q0)+(c(origLinkVars)
==1)*rbinom(popSize*numLinkVars, 1, Q1), popSize, numLinkVars);

blockids = c(t(matrix(rep(c(1:numBlocks), blockSize), numBlocks, blockSize)));

recidA = cbind(blockids, matrix(rep(matrix(c(1:blockSize), blockSize, 1), numBlocks), popSize, 1));
recidB = recidA;

#-----#
# Apply a random permutation to B records

```

```

#-----#

# Shuffle the linkage variables and the responses
# within each block

oRecidB = recidB;

matchMatrices = list();

for(b in 1:numBlocks) {

  startIndex = (b-1)*blockSize+1;
  endIndex = b*blockSize;
  permMat = diag(rep(1,blockSize));

  shuffleYes = rbinom(1,1,SHUFFLEPROBA);

  if(shuffleYes) {
      permMat = randomPermutation(blockSize);
      oRecidB[startIndex:endIndex,1:2] = permMat%%recidB[startIndex:endIndex,1:2];
      recordedLinkVarsB[startIndex:endIndex,1:numLinkVars] = permMat%%recordedLinkVarsB[startIndex:endIndex,1:numLinkVars];

      survivalTimes[startIndex:endIndex] = permMat%%survivalTimes[startIndex:endIndex];
      censored[startIndex:endIndex] = permMat%%censored[startIndex:endIndex];
      y[startIndex:endIndex] = permMat%%y[startIndex:endIndex];
      # missing_y[startIndex:endIndex] = permMat%%missing_y

  }

  matchMatrices[[b]] = permMat;

}

FPData = list(
    numBlocks = numBlocks,
    blockSize = blockSize,
    popSize = popSize,
    numLinkVars = numLinkVars,
    blocks = blocks,
    recidA = recidA,
    oRecidB = oRecidB,
    recidB = recidB,
    matchMatrices = matchMatrices,
    origLinkVars = origLinkVars,
    recordedLinkVarsA = recordedLinkVarsA,
    recordedLinkVarsB = recordedLinkVarsB,
    shuffleProba = SHUFFLEPROBA,
    p = P,
    q0 = Q0,

```

```

        q1 = Q1,
        mixingProportion = mixingProportion,
        mAgree = mAgree,
        uAgree = uAgree,
        x = x,
        y = y,
        # missing-x = missing-x,
        # missing-y = missing-y,
        # p-missing-x=p-missing-x,
        # p-missing-y=p-missing-y,
        survivalTimes = survivalTimes,
        followupTime = followupTime,
        censored = censored,
        model = 'Survival PHM',
        alpha = ALPHA,
        beta = BETA);

    return(FPData);
}

#-----#
# compareLinkVars(v1, v2)
#-----#

compareLinkVars = function(v1, v2) {

    numLinkVars = dim(v1)[2];
    numPairs = dim(v1)[1]

    gammas = matrix(rep(0,numLinkVars*numPairs),numPairs,numLinkVars);

    for(i in 1:numPairs)
        for(j in 1:numLinkVars) gammas[i,j] = (v1[i,j]==v2[i,j]);

    return(gammas);
}

#-----#
# generatePairs(FPData)
#-----#

generatePairs = function(FPData)
{

    popSize      = FPData$popSize;
    numBlocks    = FPData$numBlocks;
    blockSize    = FPData$blockSize;
    numLinkVars  = FPData$numLinkVars;
    recidA       = FPData$recidA;
    recidB       = FPData$recidB;
    oRecidB      = FPData$oRecidB;
    target_fnr   = 0.05;

```

```

# censored = FPData$censored;
mixingProportion = FPData$mixingProportion;

mAgree = FPData$mAgree;
uAgree = FPData$uAgree;

shuffleProba = FPData$shuffleProba;

ncols = 2+numLinkVars;

#-----#
# tableA: with ncols columns
#       col 1: (perfect) blocking variable
#       col 2 through col ncols-1: linkage variables
#       col ncols: covariate x
#
# tableB: with ncols columns
#       col 1: (perfect) blocking variable
#       col 2 through col ncols-1: linkage variables
#       col ncols: covariate y
#-----#

tableA = cbind(FPData$blocks, FPData$recordedLinkVarsA, FPData$x);
tableB = cbind(FPData$blocks, FPData$recordedLinkVarsB, FPData$y, FPData$censored);

for(b in 1:numBlocks) {

  # Get the records in the block b in each file

  blockA=matrix(tableA[(tableA[,1]==b)], blockSize, ncols);
  blockB=matrix(tableB[(tableB[,1]==b)], blockSize, ncols+1);

  startIndex_b=(b-1)*blockSize+1;
  endIndex_b=b*blockSize;

  oRecidB_b=matrix(rep(0, blockSize), blockSize, 1);
  oRecidB_b=oRecidB[startIndex_b:endIndex_b, 2];

  #print(list(oRecidB_b=oRecidB_b));

  for(r in 1:blockSize) {

    #-----#
    # For t the block size build
    #
    #       x_r y_1
    #       x_r y_2
    #       .   .
    #       .   .
    #       .   .
    #       x_r y_t
    #-----#

```

```

analyticalVars = cbind(matrix(rep(blockA[r,ncols],blockSize),blockSize,1), blockB[,ncols:(
ncols+1)]);

#-----#
# For t the block size build
# v_j the vector of linkage
# variables for record j
#
# gamma(v_r,v_1)
# gamma(v_r,v_2)
# . .
# . .
# . .
# gamma(v_r,v_t)
#-----#

gammas = compareLinkVars(t(matrix(rep(blockA[r,2:(ncols-1)],blockSize),numLinkVars, blockSize)
), blockB[,2:(ncols-1)]);

#-----#
# 1) block no
# 2) recid A
# 3) recid B
# 4) to (3+numLinkVars) gammas
# through gamma_K
#
# (4+numLinkVars) to
# (6+numLinkVars) x, y and censored
#
# (7+numLinkVars) to
# (11+numLinkVars) m- and u- probas
# linkage weight, cmp and lambda
#
# 12+numLinkVars match status
#-----#

tmpMat = cbind(rep(b,blockSize), rep(r,blockSize), c(1:blockSize), gammas, analyticalVars,
matrix(rep(0,5*blockSize),blockSize,5), 1*(oRecidB_b==r));

if(b==1 && r==1) potentialPairs_0=tmpMat else potentialPairs_0=rbind(potentialPairs_0, tmpMat)
;
}

}

#-----#
# E-M algorithm
#-----#

numPairs = numBlocks*blockSize^2;
pairsGammas = potentialPairs_0[,4:(4+numLinkVars-1)];
estParams = EMAlgorithm(numLinkVars=numLinkVars, blockSize=blockSize, pairsGammas=pairsGammas);

```



```

    lambda = estParams$lambda;
    m_probas = estParams$m_probas;
    u_probas = estParams$u_probas;

    m_gamma = rep(1,numPairs);
    u_gamma = rep(1,numPairs);

    for(k in 1:numLinkVars) {
        m_gamma = m_gamma*(m_probas[k]^pairsGammas[,k])*((1-m_probas[k])^(1-pairsGammas[,k]));
        u_gamma = u_gamma*(u_probas[k]^pairsGammas[,k])*((1-u_probas[k])^(1-pairsGammas[,k]));
    }

    w_gamma = log(m_gamma/u_gamma);
    q_gamma = lambda*m_gamma/(lambda*m_gamma+(1-lambda)*u_gamma);

    potentialPairs_0[,7+numLinkVars] = m_gamma;
    potentialPairs_0[,8+numLinkVars] = u_gamma;
    potentialPairs_0[,9+numLinkVars] = w_gamma;
    potentialPairs_0[,10+numLinkVars] = q_gamma;
    potentialPairs_0[,11+numLinkVars] = lambda;

    result=determineThreshold(numLinkVars, estParams, target_fnr)
    threshold=result$threshold

    potentialPairs=cbind(potentialPairs_0, (potentialPairs_0[,8+numLinkVars])>=threshold))

#-----#
# Greedy linkage
#-----#

linkMatrices = list();

for(b in 1:numBlocks) {

    # Get the records in the block b in each file

    blockA=matrix(tableA[(tableA[,1]==b)], blockSize, ncols);
    blockB=matrix(tableB[(tableB[,1]==b)], blockSize, ncols+1);

    selectionSofar = c();

    for(r in 1:blockSize) {

        startIndex = (b-1)*blockSize^2+(r-1)*blockSize+1;
        endIndex = (b-1)*blockSize^2+r*blockSize;
        w_gamma = potentialPairs[startIndex:endIndex,8+numLinkVars];

        tmpMat0 = cbind(matrix(c(1:blockSize), blockSize, 1), c(w_gamma));
        tmpMat1 = matrix(tmpMat0[!(tmpMat0[,1] %in% selectionSofar),], ncol=2);

        max_w = max(tmpMat1[,2]);
        candidates = matrix(tmpMat1[(tmpMat1[,2]==max_w),], ncol=2);
        num_candidates = dim(candidates)[1];
    }
}

```

```

for(t in 1:num_candidates) {

    q = 1/(num_candidates-t+1);
    draw = rbinom(1,1,q);

    if(draw==1) {
        linkedRecidB = candidates[t,1];
        break;
    }

}

selectionSofar = c(selectionSofar, linkedRecidB);

if(r==1) linkMatrix = matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1)
else linkMatrix = cbind(linkMatrix, matrix((c(1:blockSize)==linkedRecidB)*1,blockSize,1));

}

linkMatrices[[b]] = linkMatrix;

}

#-----#
# Final output
#-----#

pairsData = list(
    numLinkVars = numLinkVars,
    mixingProportion = Fpdata$mixingProportion,
    lambda = lambda,
    m_probas = m_probas,
    u_probas = u_probas,
    linkMatrices = linkMatrices,
    potentialPairs = potentialPairs);

return(pairsData);

}

#-----#
# all gammas: a recursive function
#-----#

generateAllGammas = function (k)
{
    if(k==1) return(c(0,1))
    else {
        prevGammas = generateAllGammas(k-1);
        allGammas = rbind(matrix(rep(prevGammas,2),k-1,2^k),c(rep(0,2^(k-1)),rep(1,2^(k-1))));
        return(allGammas);
    }
}

```

```

}

#-----#
# E-M algorithm
#-----#

EMAlgorithm = function (numLinkVars, blockSize, pairsGammas)
{
    lambda = 1/blockSize;
    initMproba = runif(1,min=0.75, max=1.0);
    initUproba = runif(1,min=0.0, max=0.25);

    m_probas = matrix(rep(initMproba, numLinkVars), 1, numLinkVars);
    u_probas = matrix(rep(initUproba, numLinkVars), 1, numLinkVars);

    dFrame = as.data.frame(pairsGammas);
    freqTable1 = as.data.frame(ftable(dFrame));
    freqTable2 = freqTable1[(freqTable1$Freq > 0),];

    numProfiles=nrow(freqTable2);

    for(col in 1:numLinkVars) {

        if(col==1) {
            profilesFreqs = matrix(c(freqTable2[,col])-1, numProfiles, 1);
        }
        else {
            profilesFreqs = cbind(profilesFreqs, c(freqTable2[,col])-1);
        }
    }

    profilesFreqs = cbind(profilesFreqs, c(freqTable2$Freq), rep(1, numProfiles), rep(1, numProfiles),
        rep(0, numProfiles));

    numIter = 100;

    for(iter in 1:numIter) {

        estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

        profilesFreqs[,2+numLinkVars] = rep(1, numProfiles);
        profilesFreqs[,3+numLinkVars] = rep(1, numProfiles);

        for(k in 1:numLinkVars) {
            profilesFreqs[,2+numLinkVars] = profilesFreqs[,2+numLinkVars]*(m_probas[k]^profilesFreqs[,k])
                *((1-m_probas[k])^(1-profilesFreqs[,k]));
            profilesFreqs[,3+numLinkVars] = profilesFreqs[,3+numLinkVars]*(u_probas[k]^profilesFreqs[,k])
                *((1-u_probas[k])^(1-profilesFreqs[,k]));
        }

        profilesFreqs[,4+numLinkVars] = lambda*profilesFreqs[,2+numLinkVars]/(lambda*profilesFreqs[,2+
            numLinkVars]+(1-lambda)*profilesFreqs[,3+numLinkVars]);
    }
}

```

```

for(k in 1:numLinkVars) {
  m_probas[k] =sum(profilesFreqs[,k]*profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars])
    /sum(profilesFreqs[,4+numLinkVars]*profilesFreqs[,1+numLinkVars]);
  u_probas[k] =sum(profilesFreqs[,k]*(1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+
    numLinkVars])/sum((1-profilesFreqs[,4+numLinkVars])*profilesFreqs[,1+numLinkVars]);
}

}

estParams = list(lambda=lambda, m_probas=m_probas, u_probas=u_probas);

return(estParams);

}

#-----#
# determineThreshold:
#-----#

determineThreshold = function (numLinkVars, rlParams, target_fnr)
{
  allGammas<-generateAllGammas(numLinkVars)

  numProfiles<-2^numLinkVars

  lambda = rlParams$lambda;
  m_probas = rlParams$m_probas;
  u_probas = rlParams$u_probas;

  m_gamma = rep(1,numProfiles);
  u_gamma = rep(1,numProfiles);

  for(k in 1:numLinkVars) {
    m_gamma = m_gamma*(m_probas[k]^allGammas[k,])*((1-m_probas[k])^(1-allGammas[k,]));
    u_gamma = u_gamma*(u_probas[k]^allGammas[k,])*((1-u_probas[k])^(1-allGammas[k,]));
  }

  w_gamma = log(m_gamma/u_gamma);

  weight_order<-order(w_gamma);
  sum_m=m_gamma[weight_order[1]];

  t=1;
  while(sum_m<=target_fnr && t<numProfiles){
    t=t+1
    sum_m=sum_m+m_gamma[weight_order[t]]
  }
  if(t==1) threshold=w_gamma[weight_order[1]]
  else if(sum_m>target_fnr && t>1) threshold=w_gamma[weight_order[t-1]]
  else threshold=w_gamma[weight_order[t]]

  est_fnr=sum(m_gamma*(w_gamma<threshold))

```

```

    est_fpr=sum(u_gamma*(w_gamma>=threshold))

    result=list(threshold=threshold, est_fnr=est_fnr, est_fpr=est_fpr)

    return(result)
}

#-----#
# Performance measures
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfOne = function(beta_0, beta_1, all_estimates, scen, estMethod) {

    selectedscen=subset(all_estimates, (all_estimates[,1]==scen));
    selectedEstimates=subset(selectedscen, (selectedscen[,3]==estMethod));

    numRows=nrow(selectedEstimates);
    bias_beta0=round(100*mean(selectedEstimates[,4]-beta_0)/beta_0,3);
    mse_beta0=round(mean((selectedEstimates[,4]-beta_0)^2),6);
    var_beta0=round((sum((selectedEstimates[,4]-mean(selectedEstimates[,4]))^2)/(numRows-1)),6);

    bias_beta1=round(100*mean(selectedEstimates[,5]-beta_1)/beta_1,3);
    mse_beta1=round(mean((selectedEstimates[,5]-beta_1)^2),6);
    var_beta1=round((sum((selectedEstimates[,5]-mean(selectedEstimates[,5]))^2)/(numRows-1)),6);

    result=rbind(c(scen,estMethod,bias_beta0,var_beta0,mse_beta0),c(scen,estMethod,bias_beta1,
        var_beta1,mse_beta1));
    return(result)
}

#-----#
# Performance measure for all methods
# cmp method beta0 bias se mse
# cmp method beta1 bias se mse
#-----#

perfAll = function(beta_0, beta_1, all_estimates, numScen) {

    for(scen in 1:numScen) {
        if(scen==1) allResults=perfOne(beta_0, beta_1, all_estimates,scen,1)
        else allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,1));
        allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,3))
        allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,4))
        allResults=rbind(allResults, perfOne(beta_0, beta_1, all_estimates,scen,2))
    }

    for(t in 1:4) {
        if(t==1) finalResults=allResults[1,]
        else finalResults=rbind(finalResults, allResults[2*(t-1)+1,])
    }
    for(t in 1:4) finalResults=rbind(finalResults, allResults[2*t,]);
}

```

```

    return(finalResults)
}

#-----#
# Function: survival score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
# scoreOption:
#         4: complete data for survival
#         5: naive QL for survival
#-----#

# survivalScore = function(beta, pairs, minCmp, scoreOption) {
#
#     numPairs=nrow(pairs);
#     totalScore=0;
#     beta_0=beta[1];
#     beta_1=beta[2];
#
#     for(r in 1:numPairs){
#         x_hi=pairs[r,1];
#         z_hj=pairs[r,2];
#         censored_hj=pairs[r,7];
#         m_hij=pairs[r,3];
#         l_hij=(pairs[r,4]>=minCmp);
#         eta_hi=beta_0+beta_1*x_hi;
#         log_f_ij=((scoreOption==0)*m_hij+(scoreOption==1)*l_hij)*((1-censored_hj)*(eta_hi-exp(
#             eta_hi)*z_hj) + censored_hj*(-exp(eta_hi)*z_hj));
#         totalScore=totalScore+log_f_ij;
#     }
#     return(totalScore);
# }

survivalScore = function(beta, pairs, minCmp, scoreOption) {

    numPairs=nrow(pairs);
    totalScore=0;
    beta_0=beta[1];
    beta_1=beta[2];

    followupTime=2.0

    for(r in 1:numPairs){
        x_hi=pairs[r,1];
        z_hj=pairs[r,2];
        censored_hj=pairs[r,7];
        m_hij=pairs[r,3];
        l_hij=(pairs[r,4]>=minCmp);
        eta_hi=beta_0+beta_1*x_hi;

```

```

    if (censored_hj==0){
      log_f_ij=((scoreOption==0)*m_hij+(scoreOption==1)*l_hj)*((eta_hi-exp(eta_hi)*z_hj) -log(1-exp
        (-exp(eta_hi)*followupTime)));
      totalScore=totalScore+log_f_ij;
    }
  }
  return(totalScore);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
#         6: least squares PW
#-----#

PW1SurvivalScore = function(beta, followupTime, pairs, numBlocks, blockSize, minCmp) {

  totalScore=0;
  beta_0=beta[1];
  beta_1=beta[2];

  for(b in 1:numBlocks) {
    block_pairs=subset(pairs, pairs[,6]==b)
    xs=block_pairs[,1]
    etas=beta_0+beta_1*xs

    subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
    numPairs=nrow(subset_block_pairs);

    if (numPairs>0){
      for(r in 1:numPairs){
        x_hi=subset_block_pairs[r,1];
        z_hj=subset_block_pairs[r,2];
        censored_hj=subset_block_pairs[r,7];
        q_hij=subset_block_pairs[r,4];
        eta_hi=beta_0+beta_1*x_hi;
        if (censored_hj==0){
          f_ij=(1-censored_hj)*exp(eta_hi)*exp(-exp(eta_hi)*z_hj) + censored_hj*exp(-exp(
            eta_hi)*z_hj)
          all_fs=(1-censored_hj)*exp(etas)*exp(-exp(etas)*z_hj) + censored_hj*exp(-exp(etas)
            *z_hj)
          avg_f_other=(sum(all_fs)-blockSize*f_ij)/(blockSize*(blockSize-1))

          num_f_ij=exp(eta_hi)*exp(-exp(eta_hi)*z_hj)
          num_all_fs=exp(etas)*exp(-exp(etas)*z_hj)
          num_avg_f_other=(sum(num_all_fs)-blockSize*num_f_ij)/(blockSize*(blockSize-1))

```

```

        numerator=q_hij*f_ij+(1-q_hij)*avg_f_other

        denom_f_ij=1-exp(-exp(eta_hi)*followupTime)
        denom_all_fs=1-exp(etas)*exp(-exp(etas)*followupTime)
        denom_avg_f_other=(sum(denom_all_fs)-blockSize*denom_f_ij)/(blockSize*(blockSize
            -1))

        denominator=q_hij*denom_f_ij+(1-q_hij)*denom_avg_f_other

        totalScore=totalScore+log(numerator/denominator)
    }
}
}
return(totalScore);
}

#-----#
# Function: Pairwise linear score
# assume iid observations
# small linkage errors
#
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l),
#         block id (b)
# scoreOption:
#         9: PW2 least squares
#-----#

PW2SurvivalScore = function(beta, followupTime, pairs, numBlocks, blockSize, minCmp) {

    totalScore=0;
    beta_0=beta[1];
    beta_1=beta[2];

    xs=pairs[,1]
    etas=beta_0+beta_1*xs

    for(b in 1:numBlocks) {
        block_pairs=subset(pairs, pairs[,6]==b)
        subset_block_pairs=subset(block_pairs, block_pairs[,4]>=minCmp);
        numPairs=nrow(subset_block_pairs);

        if(numPairs>0){
            for(r in 1:numPairs){
                x_hi=subset_block_pairs[r,1];
                z_hj=subset_block_pairs[r,2];
                censored_hj=subset_block_pairs[r,7];
                q_hij=subset_block_pairs[r,4];
                eta_hi=beta_0+beta_1*x_hi;
                if(censored_hj==0){
                    num_f_ij=exp(eta_hi)*exp(-exp(eta_hi)*z_hj)
                    num_all_fs=exp(etas)*exp(-exp(etas)*z_hj)

```



```

        num_avg_f=mean( num_all_fs )
        numerator=q_hij*num_f_ij+(1-q_hij)*num_avg_f

        denom_f_ij=1-exp(-exp( eta_hi )*followupTime)
        denom_all_fs=1-exp(-exp( etas )*followupTime)
        denom_avg_f=mean( denom_all_fs )
        denominator=q_hij*denom_f_ij+(1-q_hij)*denom_avg_f

        totalScore=totalScore+log( numerator/denominator )
    }
}
}
}
return( totalScore );
}

#-----#
# Function: minimize score
# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
# scoreType:
#
#         0: complete data for linear and homoschedastic
#         1: naive BLUE for linear and homoschedastic
#
#         2: complete data for logistic
#         3: naive QL for logistic
#
#         4: complete data for survival
#         5: naive QL for survival
#
# 6: PW LSE linear homoschedastic
# 7: PW WLS linear homoschedastic
#-----#

maximizeScore = function( initBeta=c(0,0) , pairs , followupTime=0, numBlocks , blockSize , minCmp=0.0,
                          scoreOption ) {

    if( scoreOption==0 | scoreOption==1 ) result <- optim( initBeta , survivalScore , gr=NULL, pairs=pairs
        , minCmp=minCmp, scoreOption=scoreOption , method=c("BFGS") , control=list( fnscale=-1))
    else if( scoreOption==2 ) result <- optim( initBeta , PW1SurvivalScore , gr=NULL, followupTime=
        followupTime , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize , minCmp=minCmp , method=c(
        "BFGS") , control=list( fnscale=-1))
    else if( scoreOption==3 ) result <- optim( initBeta , PW2SurvivalScore , gr=NULL, followupTime=
        followupTime , pairs=pairs , numBlocks=numBlocks , blockSize=blockSize , minCmp=minCmp , method=c("
        BFGS") , control=list( fnscale=-1))
    return( result );
}

#-----#
# Simulation parameters
#-----#

```

```

        numBlocks = 128;
        numRepetitions = 100;
        numIter = 10;
        beta = matrix(c(0.5,1.0),2,1);

        scenarioList = list();
        ScenarioResults = list()

        scenarioList[[1]] = list(blockSize=8,numLinkVars=8);
        scenarioList[[2]] = list(blockSize=4,numLinkVars=8);
        scenarioList[[3]] = list(blockSize=8,numLinkVars=8);

        sink("output.txt");

#-----#
# Actual simulations
#-----#

#-----#
# All estimates
# for each row
# -scenario
# -iteration
# -cmp
# -method: 1-14
# -estimated beta_0
# -estimated beta_1
#-----#

for(scen in 1:1) {

        blockSize = scenarioList[[scen]]$blockSize;
        numLinkVars = scenarioList[[scen]]$numLinkVars;

        allGammas = t(generateAllGammas(numLinkVars));

        for(t in 1:numRepetitions) {

                if(t%%5==0) {
                        print(list(Iteration=t));
                }

                FPData = generateFinitePopulation(numBlocks=numBlocks, blockSize=blockSize, numLinkVars=
                        numLinkVars);

                #print(list(y=FPData$y))

#-----#

                pairsData = generatePairs(FPData=FPData);
                popSize = FPData$popSize;
                blockSize = FPData$blockSize;

```

```

numBlocks = FPData$numBlocks;
followupTime=FPData$followupTime

# pairs: matrix with covariate (x), observed response (z)
#         match status (m), cmp (q) and link status (l)
#         block (b), censored (c)

potentialPairs = pairsData$potentialPairs;
  pairs = matrix(rep(0,7*numBlocks*blockSize^2),numBlocks*blockSize^2,7);
pairs[,1] = potentialPairs[, (4+numLinkVars)];
pairs[,2] = potentialPairs[, (5+numLinkVars)];
pairs[,3] = potentialPairs[, (12+numLinkVars)];
pairs[,4] = potentialPairs[, (10+numLinkVars)];
pairs[,5] = potentialPairs[, (13+numLinkVars)];
pairs[,6] = potentialPairs[, 1];
pairs[,7] = potentialPairs[, (6+numLinkVars)];

initBeta=beta+c(rnorm(2,0,0.25));

minCmp = 0.9;

#-----#
# Naive estimator
# method 1
#-----#
scoreOption = 1;
result = maximizeScore(initBeta=initBeta, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize,
  minCmp=minCmp, scoreOption=scoreOption);
  if(scen==1 && t==1) all_estimates=c(scen, t, 1, result$par)
  else all_estimates=rbind(all_estimates, c(scen, t, 1, result$par))

#-----#
# Complete data
# method 2
#-----#
scoreOption = 0;
result = maximizeScore(initBeta=initBeta, pairs=pairs, numBlocks=numBlocks, blockSize=blockSize,
  minCmp=minCmp, scoreOption=scoreOption);
  all_estimates=rbind(all_estimates, c(scen, t, 2, result$par))

#-----#
# PW1
# method 3
#-----#
scoreOption = 2;
result = maximizeScore(initBeta=initBeta, pairs=pairs, followupTime=followupTime, numBlocks=
  numBlocks, blockSize=blockSize, minCmp=minCmp, scoreOption=scoreOption);
all_estimates=rbind(all_estimates, c(scen, t, 3, result$par))

#-----#
# PW2
# method 4
#-----#

```

```

scoreOption = 3;
result = maximizeScore(initBeta=initBeta , pairs=pairs , followupTime=followupTime , numBlocks=
    numBlocks , blockSize=blockSize , minCmp=minCmp , scoreOption=scoreOption );
all_estimates=rbind(all_estimates , c(scen , t , 4 , result$par))
}

write.csv(all_estimates , file = "survival_all_estimates_k8-Nh8-missing_cmp90.csv")

numScen=1
allResults=perfAll(beta[1] , beta[2] , all_estimates , numScen)

write.csv(allResults , file = "survivalResults_k8-Nh8-missing_cmp90.csv")

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

#}
#-----#

}

sink();

```

B.3 Chapter 5

The following SAS code was used.

```

options symbolgen;

libname local 'F:\Research\Design-based RL - Sym. 2014\simulations\data';

proc printto log='F:\Research\Design-based RL - Sym. 2014\simulations\log\design-based RL -
    simulations - log.txt' new;
run;

proc printto print='F:\Research\Design-based RL - Sym. 2014\simulations\output\design-based RL -
    simulations - output.txt' new;
run;

/*
libname local 'C:\Users\abeldasyva\Documents\Design-based RL - Sym. 2014\simulations\data\
    Scenario 5 - linear';

proc printto log='C:\Users\abeldasyva\Documents\Design-based RL - Sym. 2014\simulations\log\
    Scenario 5 - linear\design-based RL - simulations - log.txt' new;
run;

```

```

proc printto print='C:\Users\abeldasyva\Documents\Design-based RL - Sym. 2014\simulations\output\
    Scenario 5 - linear\design-based RL - simulations - output.txt' new;
run;
*/

/*-----*
 * Design-based RL simulations *
 *-----*/

%let block_size=10;
%let num_blocks=1000;
%let num_individuals=%eval(&block_size*&num_blocks);

%let num_link_variables=7;

%let p_ksi_0=0.5;
%let p_ksi_1=0.5;
%let ksi_mixture_proportion=0.5;

%let p_x=0.5;
%let p_y_given_x_0=0.4;
%let p_y_given_x_1=0.7;

%let p_c_given_ksi_0=0.01;
%let p_c_given_ksi_1=0.99;

%let p_clerical_error=0.01;

%let num_iter=100;

%let num_pairs=%eval(&block_size*&block_size*&num_blocks);

%let precision=6;

/* previously 1000 */
%let sample_size=1000;

/* previously 100 */
%let num_samples=100;

%let num_substrata=10;

/*-----*
 * Generate the *
 * registers *
 *-----*/

%macro generate_pairs();

/*
    The people
*/
data individuals;

```

```

/* */
do i=1 to &num_individuals.;
/* */
block=1+floor((i-1)/&block_size.);
/* */
ksi_class=rand('BERNOULLI',&ksi_mixture_proportion.);
%do k=1 %to &num_link_variables.;
    if ksi_class eq 0 then ksi_&k.=rand('BERNOULLI',&p_ksi_0.);
    else ksi_&k.=rand('BERNOULLI',&p_ksi_1.);
%end;
/* */
x=rand('BERNOULLI',&p_x.);
if x eq 0 then y=rand('BERNOULLI',&p_y_given_x_0.);
else y=rand('BERNOULLI',&p_y_given_x_1.);
/* */
output;
/* */
drop ksi_class;
end;
/* */
run;

/*
The 1st register
*/
data register_a;
set individuals(rename=(x=x_i));
/* */
%do k=1 %to &num_link_variables.;
    if ksi_&k. eq 0 then c_i_&k.=rand('BERNOULLI',&p_c_given_ksi_0.);
    else c_i_&k.=rand('BERNOULLI',&p_c_given_ksi_1.);
%end;
/* */
drop y %do k=1 %to &num_link_variables.; ksi_&k. %end;;
run;

/*
The 2nd register
*/
data register_b;
set individuals(rename=(i=j y=y_j));
/* */
%do k=1 %to &num_link_variables.;
    if ksi_&k. eq 0 then c_j_&k.=rand('BERNOULLI',&p_c_given_ksi_0.);
    else c_j_&k.=rand('BERNOULLI',&p_c_given_ksi_1.);
%end;
/* */
drop x %do k=1 %to &num_link_variables.; ksi_&k. %end;;
run;

/*
The pairs
*/

```

```

proc sql;
  create table pairs as select a.*, b.* from register_a as a, register_b(rename=(block=block_b)) as
    b where a.block=b.block_b;
  alter table pairs drop block_b;
quit;

%let gamma_length=%eval(2+2*&num_link_variables);

/* pairs-with-outcomes */
data pairs_with_outcomes;

  attrib gamma length=$&gamma_length.;
  attrib gamma_0 length=$2;
  %do k=1 %to &num_link_variables.;
    attrib gamma.&k. length=$1;
  %end;

  set pairs;

  /* */
  m_ij=(i=j);
  /* */
  clerical_m_ij=rand('BERNOULLI',&p_clerical_error.)*(1-2*m_ij)+m_ij;

  %do x=0 %to 1; %do y=0 %to 1;
    z_ij_&x.&y.=(x_i=&x. and y_j=&y.);
  %end; %end;

  /* Use x_i and y_j as linkage variables */
  gamma_0=cats(put(x_i,1.),put(y_j,1.));
  gamma=gamma_0;
  /* */
  %do k=1 %to &num_link_variables.;
    gamma.&k.=put((c_i_&k.=c_j_&k.),1.);
    gamma=cats(gamma,'_',gamma.&k.);
  %end;

run;

proc sql;

  %do x=0 %to 1; %do y=0 %to 1;
    title "m_0_&x&y";
    select sum((gamma_0="&x&y" and m_ij=1))/sum((m_ij=1)) from pairs_with_outcomes;
    title "u_0_&x&y";
    select sum((gamma_0="&x&y" and m_ij=0))/sum((m_ij=0)) from pairs_with_outcomes;
  %end; %end;

  %do k=1 %to &num_link_variables.;
    title "m_&k";
    select sum((gamma.&k.="1" and m_ij=1))/sum((m_ij=1)) from pairs_with_outcomes;
    title "u_&k";
    select sum((gamma.&k.="1" and m_ij=0))/sum((m_ij=0)) from pairs_with_outcomes;
  %end;

```

```

    %end;
    title;

quit;

/*
    Store the parameters in a dataset
*/

%mend;

%generate_pairs;

/*-----*
 * E-M algorithm      *
 * with C.I.          *
 *-----*/

%macro em_algorithm();

/*
    Initialization
*/

data _null_;

/* */
%do x=0 %to 1; %do y=0 %to 1;
    call symputx("m_0_&x&y",0.25);
    call symputx("u_0_&x&y",0.25);
%end; %end;

/* */
%do k=1 %to &num_link_variables.;
    call symputx("m_&k",0.9);
    call symputx("u_&k",0.1);
%end;

/* The mixing proportion */
call symputx("lambda",0.1);

run;

/*
    Dataset for the e-step
*/

proc sort data=pairs-with-outcomes;
    by gamma;
run;

proc freq data=pairs-with-outcomes noprint;
    tables gamma / out=gamma-freq;

```



```

run;

/*

*/

proc sort data=pairs_with_outcomes(keep=gamma gamma_0 %do k=1 %to &num_link_variables.; gamma.&k.
    %end;) out=unique_outcomes nodupkey;
    by gamma;
run;

proc sort data=gamma_freq;
    by gamma;
quit;

data outcomes_freq;
    merge gamma_freq(in=a) unique_outcomes(in=b);
    by gamma;
    if a and b;
run;

/* */
data local.all_params;

    iter=0;
    lambda=input(symget("lambda"),8.&precision.);

    %do x=0 %to 1; %do y=0 %to 1;
        m_0_&x.&y.=input(symget("u_0_&x&y"),8.&precision.);
        u_0_&x.&y.=input(symget("u_0_&x&y"),8.&precision.);
    %end; %end;

    %do k=1 %to &num_link_variables.;
        m_&k.=input(symget("m_&k"),8.&precision.);
        u_&k.=input(symget("u_&k"),8.&precision.);
    %end;

/* */
output;

run;

/*
    Main loop
*/

%do iter=1 %to &num_iter.;

    data e_step_data;
        set outcomes_freq;
        /* */
        lambda=input(symget("lambda"),8.&precision.);

        %do x=0 %to 1; %do y=0 %to 1;

```

```

    m_0_&x.&y.=input(symget("m_0_&x&y"),8.&precision.);
    u_0_&x.&y.=input(symget("u_0_&x&y"),8.&precision.);
%end; %end;
m_0_00=1-m_0_10-m_0_01-m_0_11;
u_0_00=1-u_0_10-u_0_01-u_0_11;

%do k=1 %to &num.link.variables.;
    m_&k.=input(symget("m_&k"),8.&precision.);
    u_&k.=input(symget("u_&k"),8.&precision.);
%end;

/* */
m_proba=1;
u_proba=1;
%do x=0 %to 1; %do y=0 %to 1;
    if gamma_0="&x&y" then do;
        m_proba=m_proba*m_0_&x.&y.;
        u_proba=u_proba*u_0_&x.&y.;
    end;
%end; %end;

%do k=1 %to &num.link.variables.;
    m_proba=m_proba*(m_&k.*(gamma_&k.="1")+(1-m_&k.)*(gamma_&k.="0"));
    u_proba=u_proba*(u_&k.*(gamma_&k.="1")+(1-u_&k.)*(gamma_&k.="0"));
%end;

/* */
conditional_match_proba=m_proba*lambda/(m_proba*lambda+u_proba*(1-lambda));
run;

/*
Update the parameters
*/
proc sql;

/* Update the mixing proportion */
select put(sum(count*conditional_match_proba)/&num.pairs.,8.&precision.) into:lambda from
    e_step_data;

/* Update m_0_xy and u_0_xy */
%do x=0 %to 1; %do y=0 %to 1;
    select put(sum(count*conditional_match_proba*(gamma_0="&x&y"))/sum(count*
        conditional_match_proba),8.&precision.) into:m_0_&x.&y. from e_step_data;
    select put(sum(count*(1-conditional_match_proba)*(gamma_0="&x&y"))/sum(count*(1-
        conditional_match_proba)),8.&precision.) into:u_0_&x.&y. from e_step_data;
%end; %end;

/* Update m_k and u_k */
%do k=1 %to &num.link.variables.;
    select put(sum(count*conditional_match_proba*(gamma_&k.="1"))/sum(count*
        conditional_match_proba),8.&precision.) into:m_&k. from e_step_data;
    select put(sum(count*(1-conditional_match_proba)*(gamma_&k.="1"))/sum(count*(1-
        conditional_match_proba)),8.&precision.) into:u_&k. from e_step_data;

```

```

%end;
quit;

/* */
proc sql;
  insert into local.all_params
    set iter=&iter.,
        lambda=input(symget("lambda"),8.&precision.)
  %do x=0 %to 1; %do y=0 %to 1;
    ,m_0_&x.&y.=input(symget("m_0_&x&y"),8.&precision.),
    u_0_&x.&y.=input(symget("u_0_&x&y"),8.&precision.)
  %end; %end;
  %do k=1 %to &num_link_variables.;
    ,m_&k.=input(symget("m_&k"),8.&precision.),
    u_&k.=input(symget("u_&k"),8.&precision.)
  %end;;
quit;

%end;

%mend;

%em_algorithm;

%macro sampling_estimation();

data _null_;
  set local.all_params(where=(iter=&num_iter.));
  call symputx('lambda',lambda);
  %do x=0 %to 1; %do y=0 %to 1;
    call symputx("m_0_&x&y",m_0_&x.&y.);
    call symputx("u_0_&x&y",u_0_&x.&y.);
  %end; %end;
  %do k=1 %to &num_link_variables.;
    call symputx("m_&k",m_&k.);
    call symputx("u_&k",u_&k.);
  %end;
run;

/*-----*
 * Totals of interest *
 *-----*/

proc sql;
  %do x=0 %to 1; %do y=0 %to 1;
    select sum(m_ij*z_ij_&x.&y.) into: total_&x.&y. from pairs_with_outcomes;
  %end; %end;
quit;

/*-----*
 * Pairs with conditional match *
 * probabilities *
 *-----*/

```

```

data pairs_with_conditional_proba;
set pairs_with_outcomes;

/* */
lambda=input(symget("lambda"),8.&precision.);
%do x=0 %to 1; %do y=0 %to 1;
  m_0_&x.&y.=input(symget("m_0_&x&y"),8.&precision.);
  u_0_&x.&y.=input(symget("u_0_&x&y"),8.&precision.);
%end; %end;
m_0_00=1-m_0_10-m_0_01-m_0_11;
u_0_00=1-u_0_10-u_0_01-u_0_11;

%do k=1 %to &num_link_variables.;
  m_&k.=input(symget("m_&k"),8.&precision.);
  u_&k.=input(symget("u_&k"),8.&precision.);
%end;

/* */
m_proba=1;
u_proba=1;
%do x=0 %to 1; %do y=0 %to 1;
  if gamma_0="&x&y" then do;
    m_proba=m_proba*m_0_&x.&y.;
    u_proba=u_proba*u_0_&x.&y.;
  end;
%end; %end;

%do k=1 %to &num_link_variables.;
  m_proba=m_proba*(m_&k.*(gamma_&k.="1")+(1-m_&k.)*(gamma_&k.="0"));
  u_proba=u_proba*(u_&k.*(gamma_&k.="1")+(1-u_&k.)*(gamma_&k.="0"));
%end;

/* */
conditional_match_proba=m_proba*lambda/(m_proba*lambda+u_proba*(1-lambda));
run;

/*-----*
 * Find the different strata sizes *
 * and sample sizes *
 *-----*/

proc sql;
%do x=0 %to 1; %do y=0 %to 1;
  select count(*) into:stratum_size_&x.&y. from pairs_with_outcomes where gamma_0="&x&y";
  select min(count(*),&sample_size.) into:sample_size_&x.&y. from pairs_with_outcomes where
    gamma_0="&x&y";
%end; %end;
quit;

/*-----*
 * Pairs with strata *
 *-----*/

```

```

data pairs_with_strata;
  set pairs_with_conditional_proba;

  stratum=gamma_0;

  %do x=0 %to 1; %do y=0 %to 1;
    if gamma_0 eq "&x&y" then do;
      stratum_size      = input(symget("stratum_size_&x&y"),8.);
      stratum_sample_size = input(symget("sample_size_&x&y"),8.);
    end;
  %end; %end;

run;

/*-----*
 * Assign the substrata      *
 * of approximately equal sizes *
 * within each stratum      *
 *-----*/

proc sort data=pairs_with_strata;
  by gamma_0 conditional_match_proba;
run;

data pairs_with_substrata;
  retain substratum_obs_id;
  retain substratum;

  /* */
  set pairs_with_strata;
  by gamma_0;

  if first.gamma_0 then do;
    substratum_obs_id = 1;
    substratum=1;
  end;
  else do;

    /* Start a new substratum */
    if substratum_obs_id ge stratum_size/num_substrata. then do;
      substratum_obs_id=1;
      substratum=substratum+1;
    end;
    /* otherwise continue in the same substratum */
    else do;
      substratum_obs_id=substratum_obs_id+1;
    end;

  end;

  /* */
  drop substratum_obs_id;

```

```

run ;

/*-----*
 * Compute the substrata sizes      *
 * and sample sizes                  *
 *-----*/

proc sql;
%do x=0 %to 1;
%do y=0 %to 1;

    /* Stratum variance */
    select sum(conditional_match_proba*(1-conditional_match_proba)) into: stratum_variance_&x.&y.
        from pairs_with_substrata where gamma_0="&x&y";

%do h=1 %to &num_substrata.;

    /* Substratum size */
    select count(*) into: substratum_size_&x.&y.&h. from pairs_with_substrata where gamma_0="&x&y"
        and substratum=&h.;

    /* Substratum variance */
    select sum(conditional_match_proba*(1-conditional_match_proba)) into: substratum_variance_&x.&y.&
        h. from pairs_with_substrata where gamma_0="&x&y" and substratum=&h.;

    /* Substratum sample size: proportional allocation */
    select min(input(symget("substratum_size_&x&y&h"),8.),max(2,ceil(stratum_sample_size*(input(
        symget("substratum_variance_&x&y&h"),8.))/(input(symget("stratum_variance_&x&y"),8.)))))
        into: substratum_ssize_&x.&y.&h. from pairs_with_substrata where gamma_0="&x&y" and
        substratum=&h.;

%end;
%end;
%end;
quit;

data pairs_with_strata_substrata;
set pairs_with_substrata;

%do x=0 %to 1;
%do y=0 %to 1;
%do h=1 %to &num_substrata.;
    if stratum eq "&x&y" and substratum=&h. then do;
        substratum_size      = input(symget("substratum_size_&x&y&h"),8.);
        substratum_ssize     = input(symget("substratum_ssize_&x&y&h"),8.);
    end;
%end;
%end;
%end;

run ;

/*-----*

```

```

* Main loop                                     *
*-----*/

data local.all-estimated-totals;
  iter=0;
  sample_design=.;
  %do x=0 %to 1;
  %do y=0 %to 1;
    total_&x.&y.=input(symget("total_&x&y"),8.);
    ht_est_&x.&y.=.;
    est1_&x.&y.=.;
    est2_&x.&y.=.;
  %end;
%end;
output;
run;

data local.all-beta-estimates;
  iter=0;
  sampling_method=.;
  beta_0=.;
  beta_1=.;
  output;
run;

%do iter=1 %to &num-samples.;

/*-----*
 * Select a 1st Stratified SRS Sample *
*-----*/

/*
  Draw the sample
*/

data tmp;
  set pairs_with_strata_substrata;
  u_sample=rand('UNIFORM');
run;

proc sort data=tmp;
  by stratum u_sample;
run;

data sample_1;
  retain stratum_obs_id;
  set tmp;
  by stratum;

  if first.stratum then stratum_obs_id = 1;
  else
    stratum_obs_id = stratum_obs_id+1;

  sample_inclusion_proba = stratum_sample_size/stratum_size;

```

```

sampling_weight          = 1/sample_inclusion_proba;

sample_indicator=(stratum_obs_id le stratum_sample_size);

run;

/*-----*
 * Select a 2nd Sample          *
 * with a further stratification *
 * based on the conditional match *
 * proba                        *
 *-----*/

data tmp;
  set pairs_with_strata_substrata;
  u_sample=rand('UNIFORM');
run;

proc sort data=tmp;
  by stratum substratum u_sample;
run;

data sample_2;
  retain substratum_obs_id;
  set tmp;
  by stratum substratum;

  if first.stratum then      substratum_obs_id = 1;
  else if first.substratum then substratum_obs_id = 1;
  else                      substratum_obs_id = substratum_obs_id+1;

  sample_inclusion_proba = substratum_ssize/substratum_size;
  sampling_weight       = 1/sample_inclusion_proba;

  sample_indicator=(substratum_obs_id le substratum_ssize);

run;

%do sample_design=1 %to 2;

/* */
data in_sample;
  set sample_2sample_design.(where=(sample_indicator=1) keep=sample_indicator clerical_m_ij
    conditional_match_proba sampling_weight);
run;

proc sql;
  select count(*) into:actual_sample_size from in_sample;
quit;

/*-----*
 * IML module to          *
 *-----*/

```



```

* compute beta_0 and beta_1
*-----*/

proc iml;

/*-----*
 * Compute the weighted sum of squares *
*-----*/

start WEIGHTED_SSQ(beta);

beta_0=beta[1];
beta_1=beta[2];

/*-----*
 * Read the sample data into an      *
 * array                             *
*-----*/

use in_sample var {clerical_m_ij
                   conditional_match_proba
                   sampling_weight
                   };

read all;

/* */
w_ssq=0;

do t=1 to &actual_sample_size.;
  w_ssq=w_ssq+sampling_weight[t]*(clerical_m_ij[t]-beta_0-beta_1*conditional_match_proba[t])**2/(
    conditional_match_proba[t]*(1-conditional_match_proba[t]));
end;

/* */
return(w_ssq);

finish WEIGHTED_SSQ;

/*-----*
 * Optimize beta                      *
*-----*/

start OPTIMIZE_BETA;

init_beta=j(2,1,0);
init_beta[2]=1;

/* */
optn      = j(1,2,.);
optn[1]    = 0;
optn[2]    = 3;

CALL NLPNMS(rc,beta,"WEIGHTED_SSQ",init_beta,optn);

```

```

result=j(1,4,0);

/* Iteration */
result[1]=&iter.;

/* Sampling method: 0=SRS */
result[2]=0;
result[3:4]=beta;
edit    local.all_beta_estimates;
append from result;

/* */
call symputx('beta_0',beta[1]);
call symputx('beta_1',beta[2]);

finish OPTIMIZE_BETA;

call OPTIMIZE_BETA;

quit;

/*-----*
 * Compute the estimates      *
 *-----*/

proc sql;
%do x=0 %to 1; %do y=0 %to 1;
  /* */
  select sum(clerical.m_ij*z_ij_&x.&y.*sampling_weight*sample_indicator) into: ht_est_&x.&y.
    from sample_&sample_design.;
  /* */
  select sum(conditional_match_proba*z_ij_&x.&y.) into: total1_&x.&y. from sample_&
    sample_design.;
  select sum(conditional_match_proba*z_ij_&x.&y.*sampling_weight*sample_indicator) into:
    est_total1_&x.&y. from sample_&sample_design.;
  /* */
  select sum(((input(symget('beta_0'),8.))+input(symget('beta_1'),8.)*conditional_match_proba)
    *z_ij_&x.&y.) into: total2_&x.&y. from sample_&sample_design.;
  select sum(((input(symget('beta_0'),8.))+input(symget('beta_1'),8.)*conditional_match_proba)*
    z_ij_&x.&y.*sampling_weight*sample_indicator) into: est_total2_&x.&y. from sample_&
    sample_design.;
%end; %end;
quit;

/*

*/
proc sql;
insert into local.all_estimated_totals
set iter=&iter., sample_design=&sample_design.
%do x=0 %to 1; %do y=0 %to 1;
  ,total_&x.&y.=input(symget("total_&x&y"),8.)

```

```

        ,ht_est_&x.&y.=input(symget("ht_est_&x&y"),8.)
    ,est1_&x.&y.=input(symget("ht_est_&x&y"),8.)+input(symget("total1_&x&y"),8.)-input(
        symget("est_total1_&x&y"),8.)
    ,est2_&x.&y.=input(symget("ht_est_&x&y"),8.)+input(symget("total2_&x&y"),8.)-input(
        symget("est_total2_&x&y"),8.)

%end; %end;;

/* */
%do x=0 %to 1;
%do y=0 %to 1;

    alter table local.all_estimated_totals add re_ht_&x.&y. num;
    alter table local.all_estimated_totals add re_est1_&x.&y. num;
    alter table local.all_estimated_totals add re_est2_&x.&y. num;

    alter table local.all_estimated_totals add sqre_ht_&x.&y. num;
    alter table local.all_estimated_totals add sqre_est1_&x.&y. num;
    alter table local.all_estimated_totals add sqre_est2_&x.&y. num;

%end;
%end;;

/* */
update local.all_estimated_totals
set
    re_ht_00      =(1-ht_est_00/total_00)
    ,re_est1_00   =(1-est1_00/total_00)
    ,re_est2_00   =(1-est2_00/total_00)
    ,sqre_ht_00   =(1-ht_est_00/total_00)**2
    ,sqre_est1_00 =(1-est1_00/total_00)**2
    ,sqre_est2_00 =(1-est2_00/total_00)**2
%do x=0 %to 1;
%do y=0 %to 1;
    %if &x.=1 or &y.=1 %then %do;
        ,re_ht_&x.&y.   =(1-ht_est_&x.&y./total_&x.&y.)
        ,re_est1_&x.&y.  =(1-est1_&x.&y./total_&x.&y.)
        ,re_est2_&x.&y.  =(1-est2_&x.&y./total_&x.&y.)
        ,sqre_ht_&x.&y.  =(1-ht_est_&x.&y./total_&x.&y. )**2
        ,sqre_est1_&x.&y. =(1-est1_&x.&y./total_&x.&y. )**2
        ,sqre_est2_&x.&y. =(1-est2_&x.&y./total_&x.&y. )**2
    %end;
%end;
%end;;

quit;

%end;

%end;

/*-----*
 * Final stats *
*-----*/

proc sort data=local.all_estimated_totals;
    by sample_design;
run;

```

```

proc means data=local.all_estimated_totals(keep = %do x=0 %to 1;
                                %do y=0 %to 1;
                                sqre_ht_&x.&y.
                                sqre_est1_&x.&y.
                                sqre_est2_&x.&y.
                                %end;
                                %end; sample_design where=(sample_design ne .))
                                noprint;

by sample_design;
output out=local.final_stats mean=%do x=0 %to 1;
                                %do y=0 %to 1;
                                sqre_ht_&x.&y.
                                sqre_est1_&x.&y.
                                sqre_est2_&x.&y.
                                %end;
                                %end;;

var %do x=0 %to 1;
    %do y=0 %to 1;
        sqre_ht_&x.&y.
        sqre_est1_&x.&y.
        sqre_est2_&x.&y.
    %end;
    %end;;

run;

%mend;

%sampling_estimation;

/*-----*
 * Reset the ODS *
 *-----*/

proc printto;
run;

```

B.4 Chapter 6

The following SAS code was used.

```

/*-----*/
/*-----*/
/* STEP 1
/*
/* Read the input files and create
/* all the potential pairs
/* To be done once

```

```

/*-----*/
/*-----*/

rsubmit;

/*-----*/
/* options mfile mprint;
/* filename mprint 'debugmac';
/*-----*/
options SYMBOLGEN;
%let CMDB_SAMPLING_FRACTION=0.02;
proc printto log='\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10-CorentinOnly\CMDB.CCHS\data\
    results\cchs-cmdb-analysis-v10-step-1-log.txt' new;
run;
proc printto print='\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10-CorentinOnly\CMDB.CCHS\data\
    results\cchs-cmdb-analysis-v10-step-1-output.txt' new;
run;
/*-----*/
/*-----*/

/*-----*/
/* Read the CCHS file
/*-----*/

data cchs_file_0;
    set local.table_a (where=(cchs_cycle=1.1) keep=postal_code sex birth_date given_name_first_std1
        surname_std1 cchs_cycle sampleid personid
        rename=(personid=old_pid));

    cchs_gname=given_name_first_std1;
    cchs_sname=surname_std1;
    cchs_sname_sound=soundex(cchs_sname);
    if prxmatch('/\d{8}/',birth_date) then do;
        cchs_dob=input(birth_date ,yymmdd8.);
        cchs_dd=weekday(cchs_dob);
        cchs_mm=month(cchs_dob);
        cchs_yy=year(cchs_dob);
    end;
    cchs_sex=(sex eq '1')+2*(sex eq '2');
    cchs_pcode=postal_code;

    personid=input(old_pid,8.);

    cchs_recid=_n_;

    keep cchs_recid cchs_gname cchs_sname cchs_sname_sound cchs_dob cchs_dd cchs_mm cchs_yy
        cchs_sex cchs_pcode sampleid personid;
    if (cchs_sex in (1 2)) and (cchs_gname ne '') and (cchs_sname ne '') and (cchs_dob ne .) and (
        cchs_pcode ne '');

run;

/*-----*/

```

```

/* Only keep CCHS      */
/* records where      */
/* smoker type is      */
/* 1, 2 or 3           */
/*-----*/
data smoking_dset;
    set hlth.hs(keep=sampleid personid smka_202 smkadsty);
    where smkadsty in (1 2 3 4 5 6);
run;
data cchs_subset; set cchs_file_0; run;
proc sql;
    create table cchs_file as
    select b.*
    from smoking_dset as a inner join cchs_subset as b on a.personid=b.personid and a.sampleid=b.
        sampleid;
quit;

/*-----*/
/* Subsample the CMDB
/*-----*/
data local.cmdb_sample;
    set local.cmdb_yr2000_2011_for_glink;
    r_num=rand('BERNOULLI',&CMDB.SAMPLING_FRACTION.);
    if r_num eq 1;
run;

/*-----*/
/* Read the CMDB file
/*-----*/
data cmdb_file;
    set local.cmdb_sample(keep=cmdb_given1 cmdb_surname cmdb_birthdate cmdb_deathdate cmdb_postal
        cmdb_sex
        rename=(cmdb_sex=old_sex cmdb_deathdate=old_deathdate));

    cmdb_gname=cmdb_given1;
    cmdb_sname=cmdb_surname;
    cmdb_sname_sound=soundex(cmdb_sname);
    if prxmatch('/\d{8}/',cmdb_birthdate) then do;
        cmdb_dob=input(cmdb_birthdate,yyymmdd8.);
        cmdb_dd=weekday(cmdb_dob);
        cmdb_mm=month(cmdb_dob);
        cmdb_yy=year(cmdb_dob);
    end;
    cmdb_sex=(old_sex eq '1')+2*(old_sex eq '2');
    cmdb_pcode=compress(cmdb_postal);
    cmdb_deathdate=input(old_deathdate,yyymmdd8.);

    cmdb_recid=_n_;

    keep cmdb_recid cmdb_gname cmdb_sname cmdb_sname_sound cmdb_dob cmdb_dd cmdb_mm cmdb_yy
        cmdb_sex cmdb_pcode cmdb_deathdate;
    if (cmdb_sex in (1 2)) and (cmdb_gname ne '') and (cmdb_sname ne '') and (cmdb_dob ne .) and (
        cmdb_pcode ne '')

```

```

                                and (cmdb_deathdate ne .) and (cmdb_deathdate ge '1jan2001'd);
run;

/*-----*/
/* Create the potential pairs
/*-----*/
proc sql;
    create table potential_pairs as
    select a.*, b.*, (compress(a.cchs_sname)=compress(b.cmdb_sname)) as gamma_1,
    (compress(a.cchs_gname)=compress(b.cmdb_gname)) as gamma_2, (a.cchs_yy=b.cmdb_yy) as gamma_3, (a.
        cchs_mm=b.cmdb_mm) as gamma_4,
    (a.cchs_pcode=b.cmdb_pcode) as gamma_5
    from cchs_file as a inner join cmdb_file as b on a.cchs_sname_sound=b.cmdb_sname_sound and a.
        cchs_dd=b.cmdb_dd and a.cchs_sex=b.cmdb_sex;
quit;

/* Assign the block ids */
proc sort data=potential_pairs out=blockids_0 nodupkey;
    by cchs_sname_sound cchs_dd cchs_sex;
run;

data blockids;
    set blockids_0;
    bid=_n_;
run;

proc sql;
    create table local.potential_pairs as
    select a.*, b.bid
    from potential_pairs as a inner join blockids as b on b.cchs_sname_sound=a.cmdb_sname_sound and b
        .cchs_dd=a.cmdb_dd and b.cchs_sex=a.cmdb_sex;
quit;

/*-----*/
/*-----*/
proc printto;
run;
endrsubmit;

/*-----*/
/*-----*/
/* end STEP 1
/*-----*/
/*-----*/

/*-----*/
/*-----*/
/* STEP 2
/*
/* Two key macros:
/* First macro: estimate linkage params
/*             and survival params
/*             for a sample of blocks

```

```

/*              and pairs
/* Second macro: To draw a bootstrap
/*              sample from the original
/*              sample
/*-----*/
/*-----*/

rsubmit;

options SYMBOLGEN;
options MPRINT;
%let MAX_TIME=%sysfunc(round(%sysvalf(%sysfunc(datdif('1jan2000'd,'31dec2011'd,'ACT/ACT'))/365),0.01));
%let NUMBOOT.SAMPLES=21;
%let MAX_ETA=30;

/*-----*/
/*-----*/
/* Macro to estimate the RL params      */
/* from profiles                        */
/*-----*/
/*-----*/
%macro estimate_rl_params(profiles_dset=, est_params_dset=);

proc sql;
    select count(*) into: num_profiles from &profiles_dset.;
quit;

proc sql;
    select sum(count) into: num_pairs from &profiles_dset.;
quit;

/* IML code begin*/
proc iml;

/*-----*/
/* log-likelihood      */
/*-----*/

start LL_FUNCTION(params);

    /* m_probabilities */
    m_1=params[1];
    m_2=params[2];
    m_3=params[3];
    m_4=params[4];

    /* u_probabilities */
    u_1=params[5];
    u_2=params[6];
    u_3=params[7];
    u_4=params[8];

```



```

/*-----*/
/*-----*/
lambda=params[9];

/*-----*/
* Read the sample data into an      *
* array                             *
/*-----*/

use &profiles_dset. var {count
                        gamma_1
                        gamma_2
                        gamma_3
                        gamma_4};

read all;

num_pairs=&num_pairs.;
num_profiles=&num_profiles.;

total_ll=0;

do i=1 to num_profiles;

    m_proba=(m_1**gamma_1[i])*((1-m_1)**(1-gamma_1[i]));
    m_proba=m_proba*(m_2**gamma_2[i])*((1-m_2)**(1-gamma_2[i]));
    m_proba=m_proba*(m_3**gamma_3[i])*((1-m_3)**(1-gamma_3[i]));
    m_proba=m_proba*(m_4**gamma_4[i])*((1-m_4)**(1-gamma_4[i]));

    u_proba=(u_1**gamma_1[i])*((1-u_1)**(1-gamma_1[i]));
    u_proba=u_proba*(u_2**gamma_2[i])*((1-u_2)**(1-gamma_2[i]));
    u_proba=u_proba*(u_3**gamma_3[i])*((1-u_3)**(1-gamma_3[i]));
    u_proba=u_proba*(u_4**gamma_4[i])*((1-u_4)**(1-gamma_4[i]));

    total_i=count[i]*log(lambda*m_proba+(1-lambda)*u_proba);

    total_ll=total_ll+total_i;
end;

return (total_ll/num_pairs);

finish LL_FUNCTION;

/*-----*/
Set the initial vector of parameters values
/*-----*/

init_params=j(9,1,0);

/* m_probabilities */
init_params[1]=0.9;
init_params[2]=0.9;
init_params[3]=0.9;
init_params[4]=0.9;

```

```

/* u_probabilities */
init_params[5]=0.1;
init_params[6]=0.1;
init_params[7]=0.1;
init_params[8]=0.1;

/* lambda */
init_params[9]=0.05;
    blc=j(2,9,.);
    min_proba=1e-7;
    max_proba=1-1e-7;
    blc[1,]=min_proba*j(1,9,1);
    blc[2,]=max_proba*j(1,9,1);

/* */
optn      = j(1,2,.);
optn[1]    = 1;
optn[2]    = 3;

CALL NLPNMS(rc,est_params,"LLFUNCTION",init_params,optn, blc);

    result=j(1,11,0);
result[1:9]=est_params;
    result[10]=LLFUNCTION(est_params);
    result[11]=rc;

edit    &est_params_dset.;
append from result;

quit;

%mend estimate_rl_params;

/*-----*/
/*-----*/
/* Macro to estimate the survival
/* params
/*-----*/
/*-----*/

%macro estimate_surv_params(pairs_dset=, rl_params_dset=, est_params_dset=);

proc sql;
    create table pairs_for_analysis_1 as
    select a.*, round(yrdif(cchs_dob,'1jan2000'd, 'ACT/ACT')) as cchs_age,
    round(datdif('1jan2000'd,cmdb.deathdate,'ACT/ACT')/365,0.01) as time_till_death,
    b.lambda, b.m_1, b.m_2, b.m_3, b.m_4, b.u_1, b.u_2, b.u_3, b.u_4
    from &pairs_dset. as a, &rl_params_dset.(where=(m_1 ne .)) as b;
quit;

data pairs_for_analysis_2;
    set pairs_for_analysis_1;

```

```

m_proba=(m_1**gamma_1)*((1-m_1)**(1-gamma_1));
m_proba=m_proba*(m_2**gamma_2)*((1-m_2)**(1-gamma_2));
m_proba=m_proba*(m_3**gamma_3)*((1-m_3)**(1-gamma_3));
m_proba=m_proba*(m_4**gamma_4)*((1-m_4)**(1-gamma_4));

u_proba=(u_1**gamma_1)*((1-u_1)**(1-gamma_1));
u_proba=u_proba*(u_2**gamma_2)*((1-u_2)**(1-gamma_2));
u_proba=u_proba*(u_3**gamma_3)*((1-u_3)**(1-gamma_3));
u_proba=u_proba*(u_4**gamma_4)*((1-u_4)**(1-gamma_4));

cmp=1/(1+(1/lambda-1)*u_proba/m_proba);

drop lambda m_1-m_4 u_1-u_4;
run;

data smoking_dset;
    set hlth.hs(keep=sampleid personid smkadsty);
run;

proc sql;
    create table pairs_for_analysis_3 as
    select (a.smkadsty ne 6) as smkadsty_recoded, b.*
    from smoking_dset as a inner join pairs_for_analysis_2 as b on a.personid=b.personid and a.
        sampleid=b.sampleid;
quit;

data pairs_for_analysis;
    set pairs_for_analysis_3;
    drop personid sampleid;
run;

/*-----*/
/* CCHS recid may not be unique
/* after resampling
/* This step is just a
/* control
/*-----*/
proc sort data=pairs_for_analysis out=test_dset(keep=bid cchs_recid) nodupkey;
    by bid cchs_recid;
run;

/* */
PROC SQL;
    create table covariates_freq as
    select cchs_age, smkadsty_recoded, count(*) as count
    from pairs_for_analysis(keep=bid cchs_recid cchs_age smkadsty_recoded) group by cchs_age,
        smkadsty_recoded;
QUIT;

PROC SQL;
    select max(cchs_age) into: MAX_AGE from covariates_freq;
QUIT;

```

```

data selected_pairs;
    set pairs_for_analysis(where=(cmp gt &CMP_THR_PCT.));
run;

/*-----*/
/* IML                                     */
/*-----*/

proc iml;

/*-----*/
/* PDF Exponential                       */
/*
    params[1]: event time
    params[2]: age
    params[3]: smoker type
    params[4]: beta_0
    params[5]: beta_age
    params[6]: beta_smk
*/
/*-----*/
start LOG_PDF_FUNCTION(params);

    /* Observed response: time */
    z_j=params[1];

    /* Age in years */
    age_i=params[2];

    /* smkadsty_i=0 if never smoked
       smkadsty_i=1 else */
    smkadsty_i=params[3];

    /* survival params*/
    beta_0=params[4];
    beta_age=params[5];
    beta_smk=params[6];

    /* pdf */
    eta=beta_0+beta_age*age_i+beta_smk*smkadsty_i;
    log_exp_pdf=eta-exp(eta)*z_j;

    return(log_exp_pdf);

finish LOG_PDF_FUNCTION;

/*-----*/
/* CDF Exponential                       */
/*
    params[1]: event time
    params[2]: age
    params[3]: smoker type
    params[4]: beta_0

```

```

    params[5]: beta_age
    params[6]: beta_smk
*/
/*-----*/
start LOG_CDF_FUNCTION(params);

/* Observed response: time */
z_j=params[1];

/* covariates */
    age_i=params[2];
    smkadsty_i=params[3];

/* survival params*/
    beta_0=params[4];
    beta_age=params[5];
    beta_smk=params[6];

/* cdf */
eta=beta_0+beta_age*(age_i)+beta_smk*smkadsty_i;

log_exp_cdf=log(1-exp(-exp(eta)*z_j));

return(log_exp_cdf);

finish LOG_CDF_FUNCTION;

/*-----*/
/* OVERALL_PDF_AVERAGE */
/*
/*
    params[1]: z_j
    params[2]: beta_0
    params[3]: beta_age
    params[4]: beta_smk
*/
/*-----*/
start OVERALL_PDF_AVERAGE(params);

use covariates_freq var {count
                        cchs_age
                        smkadsty_recoded};

read all;

    num_rows=nrow(cchs_age);
    total_count=sum(count);

    pdf_params=j(1,6,.);
    pdf_params[1]=params[1];
    pdf_params[4]=params[2];
    pdf_params[5]=params[3];
    pdf_params[6]=params[4];

    total_pdf=0;

```

```

do i=1 to num_rows;
    pdf_params[2]=cchs_age[i];
    pdf_params[3]=smkadsty_recoded[i];
    total_pdf=total_pdf + count[i]*exp(LOG_PDF_FUNCTION(pdf_params));
end;

close covariates_freq;

return(total_pdf/total_count);

finish OVERALL_PDF_AVERAGE;

/*-----*/
/* OVERALL_CDF_AVERAGE */
/*
    params[1]: z_j
    params[2]: beta_0
    params[3]: beta_age
    params[4]: beta_smk
*/
/*-----*/
start OVERALL_CDF_AVERAGE(params);

use covariates_freq var {count
                        cchs_age
                        smkadsty_recoded};

read all;

num_rows=nrow(cchs_age);
total_count=sum(count);

cdf_params=j(1,6,.);
cdf_params[1]=params[1];
cdf_params[4]=params[2];
cdf_params[5]=params[3];
cdf_params[6]=params[4];

total_cdf=0;

do i=1 to num_rows;
    cdf_params[2]=cchs_age[i];
    cdf_params[3]=smkadsty_recoded[i];
    total_cdf=total_cdf + count[i]*exp(LOG_CDF_FUNCTION(cdf_params));
end;

close covariates_freq;

return(total_cdf/total_count);

finish OVERALL_CDF_AVERAGE;

/*-----*/
/* SURVIVAL_LL2_FUNCTION()

```

```

/*
  params[1]: beta_0
  params[2]: beta_age
  params[3]: beta_smk
*/
/*-----*/

start SURVIVAL_LL2_FUNCTION(params);

use selected_pairs var {time_till_death
                        cchs_age
                        smkadsty_recoded
                        cmp};

read all;

                        num_rows=nrow(cchs_age);

                        params_pdf=j(1,6,.);
                        params_pdf[4:6]=params;

                        params_pdf_avg=j(1,4,.);
                        params_pdf_avg[2:4]=params;

                        total_ll=0;

                        max_time_till_death=&MAX.TIME.;

                        params_max_time_cdf=j(1,6,.);
                        params_max_time_cdf[4:6]=params;

                        params_max_time_cdf_avg=j(1,4,.);
                        params_max_time_cdf_avg[2:4]=params;
                        params_max_time_cdf_avg[1]=max_time_till_death;
                        cdf_average=OVERALL_CDF_AVERAGE(params_max_time_cdf_avg);

do i=1 to num_rows;

                        params_pdf[1]=time_till_death[i];
                        params_pdf[2]=cchs_age[i];
                        params_pdf[3]=smkadsty_recoded[i];
                        exp_pdf = exp(LOG_PDF_FUNCTION(params_pdf));

                        params_pdf_avg[1]=time_till_death[i];
                        pdf_average=OVERALL_PDF_AVERAGE(params_pdf_avg);

                        pdf_time=cmp[i]*exp_pdf+(1-cmp[i])*pdf_average;
                        ll_time=log(pdf_time);

/* */
                        params_max_time_cdf[1]=max_time_till_death;
                        params_max_time_cdf[2]=cchs_age[i];
                        params_max_time_cdf[3]=smkadsty_recoded[i];
                        exp_cdf = exp(LOG_CDF_FUNCTION(params_max_time_cdf));

```

```

cdf_max_time=cmp[i]*exp_cdf+(1-cmp[i])*cdf_average;
ll_max_time=log(cdf_max_time);

ll=ll_time-ll_max_time;

total_ll=total_ll+ll;

end;

close selected_pairs;

return(total_ll);

finish SURVIVAL_LL2_FUNCTION;

/*-----*/
/* Maximize the survival LL */
/*-----*/

/*beta_0, beta_age, beta_smk */
init_params=j(1,3,0);
init_params[1]=-5.0;
init_params[2]=0.0;
init_params[3]=0.0;

blc=j(4,5,.);
blc[3,]={ 1 &MAXAGE. 1 -1 &MAXETA.};
blc[4,]={ -1 -&MAXAGE. -1 -1 &MAXETA.};

optn = j(1,2,.);
optn[1] = 1;
optn[2] = 3;

CALL NLPNRA(rc,est_params,"SURVIVAL_LL2_FUNCTION",init_params,optn,blc) par={1e-8 1e-1};

/* CALL NLPNMS(rc,est_params,"SURVIVAL_LL2_FUNCTION",init_params,optn,blc) par={1e-8 1e-1}; */
/* NLPNMS previously used but numerically instable */

result=j(1,5,0);
result[1:3]=est_params;
result[4]=SURVIVAL_LL2_FUNCTION(est_params);
result[5]=rc;

edit &est_params_dset.;
append from result;

quit;

%mend estimate_surv_params;

/*-----*/
/*-----*/

```



```

/* Macro to estimate all the
/* parameters
/*-----*/
/*-----*/

%macro estimate_all_params(pairs_dset=, est_rl_params_dset=, est_surv_params_dset=);

proc sql;
  create table profiles as
  select gamma_1, gamma_2, gamma_3, gamma_4, gamma_5, count(*) as count
  from &pairs_dset. group by gamma_1, gamma_2, gamma_3, gamma_4, gamma_5;
quit;

%estimate_rl_params(profiles_dset=profiles, est_params_dset=&est_rl_params_dset.);
%estimate_surv_params(pairs_dset=&pairs_dset., rl_params_dset=&est_rl_params_dset.,
  est_params_dset=&est_surv_params_dset.);

%mend estimate_all_params;

/*-----*/
/*-----*/
/* Draw bootstrap sample */
/*-----*/
/*-----*/

%macro draw_bootstrap_sample(boot_sample_dset=,b_option=);

proc sort data=local.potential_pairs(keep=cchs_sname_sound cchs_dd bid) nodupkey out=
  distinct_blocks;
  by bid;
run;

data block_ids;
  set distinct_blocks(rename=(bid=old_bid));
run;

proc sql;
  select count(*) into: num_blocks from block_ids;
quit;

%if &b_option.=1 %then %do;
data block_repetitions;
  set block_ids;
  retain total_n_i;
  num_blocks=input(symget('num_blocks'),8.);
  if(_n_ eq 1) then do;
    n_i=rand('BINOMIAL',1/num_blocks,num_blocks);
    total_n_i=n_i;
  end;
  else do;
    if(total_n_i lt num_blocks) then do;
      n_i=rand('BINOMIAL',1/num_blocks,num_blocks-total_n_i);
      total_n_i=n_i+total_n_i;
    end;
  end;

```

```

end;
else do;
  n_i=0;
end;
end;
run;
%end;
%else %do;
data block_repetitions;
  set block_ids;
  n_i=1;
run;
%end;

data new_blockids_0;
  set block_repetitions(where=(n_i ge 1));
  do t=1 to n_i;
    output;
  end;
  keep old_bid;
run;

data new_blockids;
  set new_blockids_0;
  bid=_n_;
run;

proc sql;
  create table boot_sample_0 as
  select a.*, b.bid
  from local.potential_pairs(rename=(bid=old_bid)) as a inner join new_blockids as b on a.old_bid=b
    .old_bid;
quit;

data &boot_sample_dset.;
  set boot_sample_0;
  drop old_bid;
run;

%mend draw_bootstrap_sample;

/*-----*/
/*-----*/
/* Compute bootstrap estimates */
/*-----*/
/*-----*/

%macro compute_bootstrap_estimates(num_boot_samples=,
                                   rl_params_dset=,
                                   surv_params_dset=);

%draw_bootstrap_sample(boot_sample_dset=boot_sample_dset, b_option=0);
%estimate_all_params(pairs_dset=boot_sample_dset(where=(cchs_sex eq &SEX.)),

```

```

est_rl_params_dset=&rl_params_dset.,
est_surv_params_dset=&surv_params_dset.);

/*-----*/
/* Export the data */
/*-----*/

proc export
    data=selected_pairs dbms=xlsx outfile="&PAIRS.FILE_NAME..xlsx" replace;
run;

%do t=2 %to &num_boot_samples.;
%draw_bootstrap_sample(boot_sample_dset=boot_sample_dset, b_option=1);
%estimate_all_params(pairs_dset=boot_sample_dset(where=(cchs_sex eq &SEX.)),
    est_rl_params_dset=&rl_params_dset.,
    est_surv_params_dset=&surv_params_dset.);

%end;

%mend compute_bootstrap_estimates;

/*-----*/
/*-----*/
/* Main loop
/*-----*/
/*-----*/

/*-----*/
/* Scenario 1
/*-----*/

%let SEX=1;
%let CMP_THR 90;
%let CMP_THR_PCT=%sysfunc(cats(0,.,&CMP_THR));
%let LOG_FILE_NAME=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10_CorentinOnly\
    CMDB.CCHS\data\results\analysis_log_v10_step_2_.,&SEX,.,&CMP_THR,.txt));
%let OUT_FILE_NAME=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10_CorentinOnly\
    CMDB.CCHS\data\results\analysis_out_v10_step_2_.,&SEX,.,&CMP_THR,.txt));
%let PAIRS_FILE_NAME=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10_CorentinOnly\
    CMDB.CCHS\data\results\selected_pairs_.,&SEX,.,&CMP_THR));
%let SURV_PARAMS_DSET=%sysfunc(cats(surv_estimates,.,&SEX,.,&CMP_THR));
%let SURV_COX_PARAMS_DSET=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10
    _CorentinOnly\CMDB.CCHS\data\results\CoxPHMEstimates,.,&SEX,.,&CMP_THR));
%let SURV_EXP_PARAMS_DSET=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10
    _CorentinOnly\CMDB.CCHS\data\results\ExpPHMEstimates,.,&SEX,.,&CMP_THR));
%let SURV_WEI_PARAMS_DSET=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10
    _CorentinOnly\CMDB.CCHS\data\results\WeiPHMEstimates,.,&SEX,.,&CMP_THR));
%let RL_PARAMS_DSET=%sysfunc(cats(rl_estimates,.,&SEX,.,&CMP_THR));

proc printto log="&LOG_FILE_NAME" new; run;
proc printto print="&OUT_FILE_NAME" new; run;

data local.&RL_PARAMS_DSET.;

```

```

length m_1      8.
       m_2      8.
       m_3      8.
       m_4      8.
       u_1      8.
       u_2      8.
       u_3      8.
       u_4      8.
lambda         8.
       ll       8.
       rc       8.;

run;

data local.&SURV_PARAMS_DSET.;
length beta_0    8.
       beta_age  8.
       beta_smk  8.
       ll       8.
       rc       8.;

run;

%compute-bootstrap-estimates(num_boot_samples=&NUM_BOOT_SAMPLES.,
                             rl_params_dset=local.&RL_PARAMS_DSET.,
                             surv_params_dset=local.&SURV_PARAMS_DSET.);

/*-----*/
/* Export the data */
/*-----*/
proc export
data=local.&RL_PARAMS_DSET. dbms=xlsx outfile="\F8Prod02\SDLE_Analysis\PreprocessingImpact\10
_CorentinOnly\CMDB.CCHS\data\results\&RL_PARAMS_DSET..xlsx" replace;
run;

proc export
data=local.&SURV_PARAMS_DSET. dbms=xlsx outfile="\F8Prod02\SDLE_Analysis\PreprocessingImpact
\10_CorentinOnly\CMDB.CCHS\data\results\&SURV_PARAMS_DSET..xlsx" replace;
run;

proc printto;
run;

/*-----*/
/*-----*/

%let SEX=2;
%let CMP_THR 90;
%let CMP_THR_PCT=%sysfunc(cats(0,.,&CMP_THR));
%let LOG_FILE_NAME=%sysfunc(cats("\F8Prod02\SDLE_Analysis\PreprocessingImpact\10_CorentinOnly\
CMDB.CCHS\data\results\analysis_log_v10_step_2_",&SEX,.,&CMP_THR,.txt));
%let OUT_FILE_NAME=%sysfunc(cats("\F8Prod02\SDLE_Analysis\PreprocessingImpact\10_CorentinOnly\
CMDB.CCHS\data\results\analysis_out_v10_step_2_",&SEX,.,&CMP_THR,.txt));
%let PAIRS_FILE_NAME=%sysfunc(cats("\F8Prod02\SDLE_Analysis\PreprocessingImpact\10_CorentinOnly\
CMDB.CCHS\data\results\selected_pairs_",&SEX,.,&CMP_THR));

```

```

%let SURV_PARAMS_DSET=%sysfunc(cats(surv_estimates,_,&SEX,_,&CMP_THR));
%let SURV_COX_PARAMS_DSET=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10
_CorentinOnly\CMDB.CCHS\data\results\CoxPHMEstimates,_,&SEX,_,&CMP_THR));
%let SURV_EXP_PARAMS_DSET=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10
_CorentinOnly\CMDB.CCHS\data\results\ExpPHMEstimates,_,&SEX,_,&CMP_THR));
%let SURV_WELP_PARAMS_DSET=%sysfunc(cats(\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10
_CorentinOnly\CMDB.CCHS\data\results\WeiPHMEstimates,_,&SEX,_,&CMP_THR));
%let RL_PARAMS_DSET=%sysfunc(cats(rl_estimates,_,&SEX,_,&CMP_THR));

proc printto log="&LOG_FILE_NAME" new; run;
proc printto print="&OUT_FILE_NAME" new; run;

data local.&RL_PARAMS_DSET.;
length m_1 8.
       m_2 8.
       m_3 8.
       m_4 8.
       u_1 8.
       u_2 8.
       u_3 8.
       u_4 8.
lambda 8.
       ll 8.
       rc 8.;
run;

data local.&SURV_PARAMS_DSET.;
length beta_0 8.
       beta_age 8.
       beta_smk 8.
       ll 8.
       rc 8.;
run;

%compute_bootstrap_estimates(num_boot_samples=&NUM_BOOT_SAMPLES.,
                             rl_params_dset=local.&RL_PARAMS_DSET.,
                             surv_params_dset=local.&SURV_PARAMS_DSET.);

/*-----*/
/* Export the data */
/*-----*/
proc export
data=local.&RL_PARAMS_DSET. dbms=xlsx outfile="\\F8Prod02\SDLE_Analysis\PreprocessingImpact\10
_CorentinOnly\CMDB.CCHS\data\results\&RL_PARAMS_DSET..xlsx" replace;
run;

proc export
data=local.&SURV_PARAMS_DSET. dbms=xlsx outfile="\\F8Prod02\SDLE_Analysis\PreprocessingImpact
\10_CorentinOnly\CMDB.CCHS\data\results\&SURV_PARAMS_DSET..xlsx" replace;
run;

/*-----*/
/*-----*/

```

```
proc printto;
run;

endrssubmit;
/*-----*/
/*-----*/
/* end STEP 2
/*-----*/
/*-----*/

signoff;
```