Approximate Marginal Likelihoods for Shrinkage Parameter Estimation in Penalized Logistic Regression Analysis of Case-Control Data

by

Siyuan Chen

B.Sc., Xi’an Jiaotong University, 2018

Project Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in the Department of Statistics and Actuarial Science Faculty of Science

© Siyuan Chen 2020
SIMON FRASER UNIVERSITY
Spring 2020

Copyright in this work rests with the author. Please ensure that any reproduction or re-use is done in accordance with the relevant national copyright legislation.
Approval

Name: Siyuan Chen

Degree: Master of Science (Statistics)

Title: Approximate Marginal Likelihoods for Shrinkage Parameter Estimation in Penalized Logistic Regression Analysis of Case-Control Data

Examination Committee: 
Chair: Jiguo Cao
Associate Professor

Brad McNeney
Senior Supervisor
Professor

Liangliang Wang
Supervisor
Assistant Professor

Jinko Graham
Internal Examiner
Professor

Date Defended: April 17, 2020
Abstract

Inference of associations between disease status and rare exposures is complicated by the finite-sample bias of the maximum likelihood estimator for logistic regression. Penalised likelihood methods are useful for reducing such bias. In this project, we studied penalisation by a family of log-F priors indexed by a shrinkage parameter $m$. We propose a method for estimating $m$ based on an approximate marginal likelihood obtained by Laplace approximation. Derivatives of the approximate marginal likelihood for $m$ are challenging to compute, and so we explore several derivative-free optimisation approaches to obtaining the maximum marginal likelihood estimate. We conduct a simulation study to evaluate the performance of our method under a variety of data-generating scenarios, and applied the method to real data from a genetic association study of Alzheimer’s disease.

**Keywords:** case-control study; penalised likelihood; log-F priors; Laplace Approximation; Alzheimer’s Disease
Dedication

Ich steh. Ich bekenne. Ich ruf.

Ein Krieger, Paul Celan
Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Dr. Brad McNeney for the continuous support of my study and research throughout the master program, and for his patience, encouragement and guidance that help me get to know and be enchanted with statistical genetics. I would also greatly appreciate the effort and time he has put into this project. Thank you for always being so supportive, kind-hearted and insightful.

I would also like to express my thanks to my examining committees, Dr. Jiguo Cao, Dr. Jinko Graham and Dr. Liangliang Wang for taking their time to participate in my defense. Many thanks go to all the staff and faculty in the Department of Statistics and Actuarial Science for their wonderful lectures, talks and generous help. Besides, I am grateful to have the opportunity to work with Dr. Celia Greenwood from Lady Davis Institute and have such an unforgettable summer.

Furthermore, I would like to thank all my lovely friends and fellow graduate students in SFU, and talented people in Greenwood’s Lab, for the time we spent together, the laugh we had and the help I received. Special thanks go to Lulu Guo, for always being a considerate friend; also Dongmeng Liu, for offering generous help when I just arrived Canada.

Finally yet importantly, I would appreciate the love, support and company from Alexander Vasilenko, Natalia Vasilenko and Isabell (Kitty). I would also express my appreciation to my parents and grandparents for their love and care throughout my life. Thank you all for bringing me endless delights, and courage for pursuing my goals.
Table of Contents

Approval ii
Abstract iii
Dedication iv
Acknowledgements v
Table of Contents vi
List of Tables viii
List of Figures ix

1 Introduction 1

2 Methodology 3
  2.1 Marginal likelihood for Shrinkage Parameter $m$  3
  2.2 Laplace Approximation  5
  2.3 Derivative-free Optimization Strategies  8
    2.3.1 Nelder-Mead  8
    2.3.2 Genetic Algorithm  9
    2.3.3 Particle Swarm Optimization  10
  2.4 Summary of Maximum Marginal Likelihood Estimator of $m$  11

3 Simulation Study 12
  3.1 Simulation Design  12
  3.2 Study 1: Optimization Methods Comparison  13
  3.3 Study 2: Dimension of Genetic Variants  13

4 Real Data Modelling 16
  4.1 Data Description  16
  4.2 Estimating Shrinkage Parameter $m$  16
5 Discussion

Bibliography

Appendix A Implementation of Laplace Approximation

Appendix B LDheatmaps for Genes in Real Data Analysis

Appendix C Code
List of Tables

Table 3.1 Study 1, simulation setting ........................................ 13
Table 3.2 Mean and variance of estimated $m$ ................................. 14
Table 3.3 Study 2, Frequency of acceptable estimation, mean and variance of estimation of $m$ .................................................... 14
Table 4.1 Genes included from ADNI-1 study ............................... 17
Table 4.2 Estimated $m$ for each gene ........................................ 17
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2.1</td>
<td>The shape of log-F distribution</td>
<td>5</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>The original posterior density for ( \beta ) and corresponding unnormalized Normal density</td>
<td>7</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>( (\hat{L}<em>{MC} - \hat{L}</em>{LA})/\hat{L}_{MC} ), dashed lines indicating ( \mp 0.3 )</td>
<td>8</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>( \log(\hat{L}<em>{MC}) ) and ( \log(\hat{L}</em>{LA}) ) for ( m = 0.5, 1, ..., 10 )</td>
<td>9</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>Study 1, estimation of ( m ) for settings of true ( m = 2, 4, 8 ) separately</td>
<td>15</td>
</tr>
<tr>
<td>Figure B.1</td>
<td>LDheatmaps for genes included in real data analysis section, using ( R^2 ) measure of LD(part 1)</td>
<td>25</td>
</tr>
<tr>
<td>Figure B.2</td>
<td>LDheatmaps for genes included in real data analysis section, using ( R^2 ) measure of LD(part 2)</td>
<td>26</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The case-control design is common for genetic epidemiology studies of the relationship between disease status and genetic variants of interest. Case-control studies are retrospective in the sense that subjects are sampled based on disease status, with covariates such as genetic variants measured retrospectively. Thus, disease status, $Y$, is fixed and covariate information, $X$, is random. However, inference of disease-risk parameters is by logistic regression. A prospective model which assumes disease status, $Y$, is random and covariate information $X$ is fixed. The maximum likelihood estimator of the regression parameters is thus obtained under prospective sampling with fixed covariates and random binary response variables.

Regardless of the sampling design, the logistic regression estimator is known to be biased away from zero [5, 6], and the bias is particularly acute when the covariate data are sparse (e.g., many zeroes) or the sample size is small. In the case of extremely sparse or small samples the likelihood may be monotone, so that the maximum likelihood estimator does not exist.

Firth [5] proposed a bias-reduced modification of logistic regression, obtained by maximizing the logistic regression likelihood penalized by the Jeffreys prior. For regression parameters $\beta$, the Jeffreys prior is proportional to $\sqrt{|I(\beta)|}$, where $I(\beta)$ is the Fisher information matrix. Thus, Firth’s estimator maximizes the penalized log-likelihood

$$l^*(\beta) = l(\beta) + \frac{1}{2} \log |I(\beta)|$$  \hspace{1cm} (1.1)

where $l(\beta)$ is the logistic regression log-likelihood. Firth showed that the maximizer of $l^*(\beta)$ always exists and has reduced the first-order bias compared to the maximum likelihood estimator. Though there is no formal justification for using Firth logistic regression on case-control data, it has been shown to perform well in simulation studies [6].
Inspired by Firth, Zhang [28] developed a bias-reduced estimator for case-control data. Zhang’s estimator is the maximizer of a penalized profile likelihood, where the profile likelihood is obtained by maximizing over the infinite-dimensional parameter semiparametric case-control likelihood, and the penalty is like that in the equation (1.1), but with an estimate, \( \hat{I}(\theta) \), of the Fisher information in the profile likelihood. Simulation studies suggest that the Zhang’s method and Firth logistic regression have similar statistical properties when applied to case-control data [6]. The similarity of Firth and Zhang logistic regression suggests that we can apply other penalized logistic regression methods to case-control data.

Alternatives to Firth logistic regression were considered by Greenland and Mansournia[7]. They recommend penalization by log-F prior distributions over other possible priors such as normal, t- and Cauchy distributions. The family of log-F priors is indexed by a shrinkage parameter \( m \). Larger values of \( m \) induce greater shrinkage. Graham et al.[6] found that the log-F priors performed well in limited simulations of case-control data, but did not propose a method for choosing \( m \).

Greenland and Mansournia suggested an empirical Bayes approach to estimating \( m \). The empirical Bayes estimator is the maximizer of the marginal likelihood for \( m \) obtained by integrating \( \beta \) out of the joint distribution of the data and \( \beta \). Yu [27] followed this suggestion and applied the EM algorithm to maximize the marginal likelihood, treating the regression coefficients \( \beta \) as missing data. However, the integral required for the E-step is challenging, and Yu implemented a Monte Carlo EM algorithm. The Monte Carlo EM algorithm was found to be computationally slow. In this project we investigate a computationally faster alternative, based on a Laplace approximation to the marginal likelihood.

The project is organized as follows. In Chapter 2 we formulate the marginal likelihood for \( m \). Next we introduce the Laplace approximation of the marginal likelihood function and optimization methods for maximizing this approximate likelihood to obtain empirical Bayes estimates of \( m \). In Chapter 3 we provide simulation studies to evaluate our method. Chapter 4 gives an application to real data from a genetic association study of Alzheimer’s disease. Concluding remarks are made in Chapter 5.
Chapter 2

Methodology

2.1 Marginal likelihood for Shrinkage Parameter \( m \)

Consider a case-control study of association between a genetic variant and a disease status. Under case-control sampling design, cases and controls are selected from the population, and their covariates are recorded. Denote disease status on individual \( i \) as \( Y_i \in \{0, 1\} \), with 1 indicating case and 0 indicating control. Denote the \( k_{th} \) genetic variant covariate as \( X_{ki} \), \( i = 1, ..., n, k = 1, ..., K \). For each genetic variant, assume a logistic regression model with log-OR parameter \( \beta_k \) and an intercept term \( \alpha_k \):

\[
\log \left[ \frac{P(Y_i = 1 | X_{ki}^k)}{P(Y_i = 0 | X_{ki}^k)} \right] = \alpha_k + \beta_k X_{ki}^k,
\]

or, equivalently,

\[
P(Y_i = 1 | X_{ki}^k) = \frac{\exp(\alpha_k + \beta_k X_{ki}^k)}{1 + \exp(\alpha_k + \beta_k X_{ki}^k)}, \quad i = 1, ..., n; k = 1, ..., K \quad (2.1)
\]

For future reference we note that the intercept terms, \( \alpha_k \), are different for each covariate. Qin and Zhang [17] derived the following two-sample semi-parametric model for the covariate data:

\[
P(X_{ki}^k | Y_i = 0) = g(X_{ki}^k)
\]

\[
P(X_{ki}^k | Y_i = 1) = c(\beta_k, g) \exp(X_{ki}^k \beta_k)g(X_{ki}^k), \quad (2.2)
\]

where \( g() \) is the distribution of \( X^k \) in controls and \( c(\beta, g) \) is a normalization constant. The likelihood is then

\[
L(\beta, g) = \prod_{Y_i=0} g(X_{ki}^k) \prod_{Y_i=1} c(\beta_k, g) \exp(X_{ki}^k \beta_k)g(X_{ki}^k). \quad (2.3)
\]
Finding the MLE of $\beta_k$ is complicated by the infinite-dimensional nuisance parameter $g$. Qin and Zhang [17] proved that a profile likelihood function obtained by profiling $g$ out has the same form as the logistic regression (2.1), with a different intercept term $\alpha_k^*$. 

$$L(\alpha_k^*, \beta_k) = \prod_{i=1}^{n} \frac{\exp(Y_i(\alpha_k^* + X_i^k \beta_k))}{1 + \exp(\alpha_k^* + X_i^k \beta_k)}$$  \hspace{1cm} (2.4)$$

The intercept $\alpha_k^*$ can be shown to depend on the logistic regression intercept, $\alpha$, the ratio of cases to controls in the study, and the disease prevalence [20]. The dependence on $\alpha_k$ means that the intercepts are different for each covariate.

We penalize this profile likelihood with the log-$F(m, m)$ prior distribution [11]

$$p(\beta_k|m) = \frac{1}{\text{Beta}(m/2, m/2)} \frac{\exp(-m/2 \beta_k)}{(1 + \exp(-\beta_k))^m}, k = 1, ..., K$$  \hspace{1cm} (2.5)$$

The distribution has a symmetrical bell shape with mean zero and a variance that decreases as the parameter $m$ increases. When $m > 10$, the distribution is nearly Normal [7].

Now assume $K$ independent covariates. We combine the data across all $k$ to form the joint penalized likelihood

$$L(\alpha^*, \beta, m) = \prod_{i=1}^{K} L(\alpha_k^*, \beta_k)p(\beta_k|m)$$  \hspace{1cm} (2.6)$$

where $\alpha^* = (\alpha_1^*, ..., \alpha_K^*)$. A marginal likelihood for the hyper-parameter $m$ can be estimated by integrating $\beta$ out of this joint likelihood:

$$L(\alpha^*, m) = \int \prod_{i=1}^{K} L(\alpha_k^*, \beta_k)p(\beta_k|m)d\beta = \prod_{k=1}^{K} \int L(\alpha_k^*, \beta_k)p(\beta_k|m)d\beta_k$$  \hspace{1cm} (2.7)$$

The vector of intercepts $\alpha^*$ is considered to be a nuisance parameter in this marginal likelihood. We note that the dimension of this intercept increases with the number of covariates $K$.

We make two additional remarks. First, although $L(\alpha_k^*, \beta_k)$ is a penalized profile likelihood, rather than a standard likelihood function, it is of the same form as a penalized prospective likelihood of logistic models. Prentice and Pyke showed that the MLE of $\beta_k$ can be obtained by maximizing $L(\alpha_k^*, \beta_k)$ [16]. A similar approach in the context of partly linear regression models was taken by Shen [22]. Second, we note that $m$ controls the variance of the log-$F$ distribution, with larger $m$ leading to larger penalties on $\beta_k$ values that are far away from 0. In what follows we label the prior distributions as "flat", "medium" or "pointed" for $m$ $m \leq 3$, $3 \leq m \leq 7$, and $m > 7$ respectively. In the Chapter 3 simulations we choose one $m$ from each category.
Note that evaluating the integrals in $L(\alpha^*, m)$ analytically is challenging. A straight-forward way to solve this problem is to apply numerical approximate integration methods such as Gaussian approximation, Monte Carlo integration and quadrature methods. We considered Laplace Approximation in particular, because it is widely used for approximating marginal likelihoods and its simplicity in computation and minimal computation time are advantages over quadrature and MC integration, respectively.

![Figure 2.1: The shape of log-F distribution](image)

### 2.2 Laplace Approximation

Our goal is to approximate the integrals in the marginal likelihood of equation (2.7). Each integral can be viewed as the marginal distribution of the data in a Bayesian problem with likelihood $L(\alpha_k^*, \beta_k)$ and prior $p(\beta_k|m)$. We first discuss theoretical results from the Bayesian inference literature that justify Laplace approximations of marginal distributions when the sample size is large. We then present an empirical investigation of the utility of Laplace approximation in the kinds of small sample problems that we are interested in.

Laplace’s method approximates an integral by approximating the integrand with an easy-to-integrate function. In particular, the integrand is approximated by an unnormalized
Guassian density function whose mean coincides with the mode of the integrand. Suppose our integrand is an un-normalized posterior density \( P(\theta) \) with \( l(\theta) = \log P(\theta) \). If \( l(\theta) \) is a smooth function with continuous second derivative and a sharp peak about its maximum at \( \hat{\theta} \), we can approximate it with a 2-term Taylor expansion:

\[
l(\theta) \approx l(\hat{\theta}) + (\hat{\theta} - \theta)l'(\hat{\theta}) + \frac{1}{2}(\hat{\theta} - \theta)^2 l''(\hat{\theta})
\]

where \( l' \) and \( l'' \) denote the first and second derivatives of \( l \), respectively. Up to a constant, the approximation matches the log-pdf of a Gaussian distribution with mean \( \hat{\theta} \) and variance equal to the inverse of \( -l''(\hat{\theta}) \). Thus we can approximate \( P(\theta) \) by an unnormalized Normal distribution \( P(\hat{\theta}) \exp\left[{-\frac{c_P(\theta - \hat{\theta})^2}{2}}\right] \), where \( c_P = -\frac{d^2\log(P(\theta))}{d\theta^2}\big|_{\theta=\hat{\theta}} \). It follows that the desired normalization constant \( \int P(\theta)d\theta \) can be approximated by

\[
\int P(\hat{\theta})\exp\left[{-\frac{c_P(\theta - \hat{\theta})^2}{2}}\right]d\theta = P(\hat{\theta})\int \exp\left[{-\frac{c_P(\theta - \hat{\theta})^2}{2}}\right]d\theta
\]

\[
= P(\hat{\theta})\sqrt{\frac{2\pi}{c_P}}
\]

An expression for \( P(\hat{\theta})\sqrt{\frac{2\pi}{c_P}} \) in our problem is given in Appendix A.

The quality of the Laplace approximation to the posterior density function has been discussed theoretically for parametric models. For example, Tierney and Kadane [24] proved that for a smooth prior distribution and fixed value of the data the Laplace approximation to the marginal posterior density has an error that is \( O(n^{-1}) \) in a multi-parameter setting.

Another motivation for the approximation is a result called the Bernstein-von Mises Theorem (see the version below taken from [25]) that tells us that posterior distributions tend toward Gaussian distributions as the sample size increases. From this it follows that the unnormalized posterior distribution \( P(\theta) \) tends towards an unnormalized Gaussian.

**Theorem (Bernstein-von Mises Theorem)** Consider \( X_1, X_2, ..., X_n \) are IID from the pdf \( f(x_i|\theta), \theta \in \Theta \). If \( log f(x_i|\theta) \) is twice continuously differentiable w.r.t \( \theta \) for each \( x_i \), and the sample size \( n \) is large enough, for any positive, bounded and twice differentiable over \( \Theta \) prior density \( \xi(\theta) \), we have:

\[
sup_{z} \left| P(\theta \leq z | X = x) - \Phi(\sqrt{c_P}(z - \hat{\theta})) \right| \approx 0 \tag{2.10}
\]

The Bernstein-von Mises Theorem proves that when the sample size is large the posterior density will be close to a normal density, which assures the quality of the Laplace approximation.
approximation. To investigate the quality of Laplace Approximation in our context we performed limited simulations to assess whether the shape of the unnormalized posterior \( L(\alpha^*_k, \beta_k)p(\beta_k|m) \) is close to a normal density function and to judge the quality of Laplace Approximation to the marginal likelihood. The simulations were conducted as described in Chapter 3, with the exception that here we used a small sample of 10 cases and 40 controls. The results are as follows.

For a single covariate simulated under \( m = 4 \) a plot of \( L(\alpha^*_1, \beta_1)p(\beta_1|m) \) for \( m = 4 \) and \( \alpha^* = -3 \) is shown in Figure 2.2, with the approximating unnormalized Gaussian distribution superposed. We can see that the posterior is unimodal (see Appendix A for a proof of unimodality) with a heavier tail than the approximating Gaussian. Overall the approximation looks reasonable for this simulated covariate.

![Figure 2.2: The original posterior density for \( \beta \) and corresponding unnormalized Normal density](image)

Next we investigate the quality of Laplace Approximation to the marginal likelihood. For each dataset, the marginal likelihood \( L(\alpha, m) = \int L(X|\beta)p(\beta|m)d\beta \) can be regarded as \( \mathbb{E}_{\beta}(L(X)|\Theta) \), in which \( \Theta \) indicates the parameter space of \( \beta \). Such an expectation can be estimated by Monte Carlo by sampling \( \beta \)'s from the prior distribution and calculating the mean of the likelihood values from each \( \beta \). The precision of such an estimate depends on the Monte Carlo sample size. In the following results we used a Monte Carlo sample size of 1 million.

We compare the approximated marginal likelihood at \( m = 4 \) and \( \alpha^* = -3 \) from Laplace Approximation, \( \hat{L}_{LA} \), to the estimation from Monte Carlo, \( \hat{L}_{MC} \), for each of 100 simulated single-covariate datasets simulated under \( m = 4 \), and calculate the relative difference \( (\hat{L}_{MC} - \hat{L}_{LA})/\hat{L}_{MC} \). In our study, the absolute relative difference is less than 0.3 in around 75% of the datasets (see Figure 2.3).
Finally, for a single dataset we compared the MC and LA estimates of the marginal likelihood of $\alpha^* = -3$ and a grid of $m = (0.5, 1, ..., 10)$. We plot the natural log of LA and MC estimates versus $m$ (see figure 2.4). We see that the argmax of the LA-approximated marginal likelihood is smaller than the argmax of the MC-approximated marginal likelihood. Whether such underestimation is typical and leads to biased estimation of $m$ is an area for future work.

2.3 Derivative-free Optimization Strategies

We maximize the approximate marginal likelihood, denoted $\tilde{L}(\alpha^*, m)$ to estimate $(\alpha^*, m)$. Calculation of derivatives of $\tilde{L}(\alpha^*, m)$ is challenging and so we opted for derivative-free optimization methods. We consider the Nelder-Mead algorithm, a genetic algorithm, and the particle swarm optimization method. The genetic algorithm and particle swarm are examples of the larger class of evolutionary algorithms. We discuss each method briefly in the following subsections. Our simulations (Chapter 3) suggested that the genetic algorithm and the particle swarm optimization method perform better in general. Throughout we let $f(x)$ denote the objective function to be maximized over $x$ in some subset $\Omega$ of $\mathbb{R}^p$.

2.3.1 Nelder-Mead

The Nelder-Mead extended simplex method is most easily described for the case $p = 2$. Starting from an initial triangle over $\Omega$ and with $f$ evaluated at each of the vertices, we pivot the triangle by reflecting the lowest-value vertex across the edge opposite, in a way that maintains the area of the triangle. We continue this pivoting process, tending towards higher values of the objective function, until the objective function value is nearly the same at all vertices of the triangle. Refinements of the algorithm allow for the triangle to be...
expanded or contracted at different iterations [13], to change the speed at which we move through \( \Omega \). For \( p \)-dimensional \( \Omega \) we replace triangles with simplexes. Though simple, the Nelder-Mead method is not guaranteed to converge, and there are multiple examples of its failure, even in two dimensions [26, 15]. The Nelder-Mead algorithm is implemented in the R function `optim()` included in the base-R `stats` package [18].

### 2.3.2 Genetic Algorithm

Genetic algorithms are stochastic search algorithms that equate values of the objective function to genetic fitness and mimic the process of natural selection of organisms to maximize this fitness measure. Starting from an initial “generation” of candidate solution vectors \( x_1, \ldots, x_N \in \Omega \), the \( x_i \)’s are allowed to “recombine” with others or “mutate” to produce the next generation, with higher-fitness \( x_i \)’s being more likely to reproduce. In our context, recombination is the exchange of coordinates between two vectors and mutation is the change of a specific coordinate value to a new value. The solution is the highest value of the objective function found before the algorithm terminates. Termination can be because there is no improvement in the fitness value for specific number of iterations, or the algorithm...
reaches a specified maximum generations. The approach was pioneered by Holland [10] and later generalized; see Corez [4] for a review. A pseudo-code implementation of a genetic algorithm is shown in Algorithm 1 [19] below. In our study we use the \textit{ga()} function from the R package \textit{GA} [21].

\begin{algorithm}
\caption{Genetic Algorithm}
\begin{algorithmic}
\State \textbf{Result:} final population $P$
\State \textbf{Input:} evaluation function $f$, control parameters set $C$;
\State \textbf{Initialization:} random initial population $P$;
\While{termination condition not satisfied}
\State evaluate current population and get the best individuals $E$ from $P$;
\State select parents set $Parents$ from $P$;
\If{crossover condition satisfied}
\State $Children \leftarrow$ crossover($Parents$);
\EndIf
\If{mutation condition satisfied}
\State $Children \leftarrow$ mutation($Parents$);
\EndIf
\State $P \leftarrow E \bigcup Children$
\EndWhile
\end{algorithmic}
\end{algorithm}

2.3.3 Particle Swarm Optimization

Particle swarm optimization (PSO) is a stochastic optimization technique proposed by Eberhart and Kennedy [12]. An initial set of vectors $x_1, \ldots, x_N$ is viewed as “particles” that can move about $\Omega$. The velocity (direction and speed) of the movements are partly random and partly influenced by values of the objective function seen previously by the particle itself and others in its neighbourhood. Various modifications of PSO, and hybrids of PSO and other modern evolutionary optimization algorithms are reviewed by Cortez [4]. Pseudo-code for the SPSO 2007 algorithm of Clerc [3] is given in Algorithm 2 [4]. In our study we use the
psoptim() function from the R package pso [2], which implements both the SPSO 2007 and SPSO 2011 algorithms of [3].

**Algorithm 2: Particle Swarm Optimization**

**Result:** Best solution \( B \)

**Input:** evaluation function \( f \), control parameters set \( C \);

**Initialization:** initial swarm \( P \), including random position and velocity for each particle;

get the best particle \( B \);

**while** termination condition not satisfied **do**

**for each particle** \( x = (s, v, p, l) \in P \) **do**

update the velocity \( v \leftarrow \text{velocity}(s, v, p, l) \);

move the particle \( s \leftarrow s + v \);

check if \( x \) is on the boundary, if so, adjust;

if \( s \) is better fitted than \( p \) then

\( \text{p} \leftarrow s \)

end

if \( s \) is better fitted than \( B \) then

\( B \leftarrow s \)

end

end

update \( l \) for all particles following given topology in \( C \);

**end**

2.4 Summary of Maximum Marginal Likelihood Estimator of \( m \)

The approximate marginal likelihood is \( \tilde{L}(\alpha^*, m) = \prod_{k=1}^{K} \tilde{E}[L|\alpha^*, m] \), where \( \tilde{E}[L|\alpha^*, m] \) is the Laplace approximation to \( \int L(\alpha^*, \beta)p(\beta|m)d\beta \). Each approximate integral \( \tilde{E}[L|\alpha^*, m] \) can be evaluated as discussed in the Appendix. To avoid underflow we take logarithms and maximize \( \tilde{l}(\alpha^*, m) = \sum_{k=1}^{K} \log \tilde{E}[L|\alpha^*, m] \). This objective function can be passed to any of the three derivative-free optimization methods discussed in Section 2.3 to obtain the estimates \((\hat{\alpha}^*, \hat{m})\).
Chapter 3

Simulation Study

Our simulation study addressed two questions:

1. Which of Nelder-Mead, the genetic algorithm (GA) or particle swarm optimization (PSO) is the best optimization method for our problem?

2. How does the number of genetic markers affect the bias and variance of our estimator of $m$?

In this chapter we describe the design of our study and the results.

3.1 Simulation Design

Under the covariate data model (2.2) and prior distribution (2.5) the parameters of the data-generating process are (i) the numbers of cases, $n_1$, and controls, $n_0$, (ii) the distribution of covariates in controls, $g$, and (iii) the precision parameter of the prior, $m$. In addition we can control the number of genetic markers $K$. In all simulations we set $n_1 = 200$, $n_0 = 800$ and $g$ to be the standard normal distribution. We chose a relatively large sample size so that the quality of the Laplace approximations would not be an issue, and used a fairly standard 1:4 ratio of cases to controls. The choice of a standard normal $g$ was for convenience. One can show that with this distribution of covariates in controls, the distribution of covariates in cases is also normal with variance 1, but with mean equal to the log-OR parameter $\beta_k$ [27]. Thus, we can generate covariates directly from their distributions, rather than the alternative of generating a large cohort and sub-sampling cases and controls. For each simulation configuration we generated 20 data sets.

For the comparison of optimization methods (question 1) we set $K = 20$ and considered $m = 2, 4$ or 6. For the investigation of the bias and variance of our estimator as a function of $K$ (question 2) we set $m = 4$ and considered $K = 10, 20, 30, 40$ or 70. In an empirical Bayes procedure, information about the hyperparameter of the prior accrues as the number
of samples from the prior increases, which in our study is as $K$ increases. We therefore expect bias and variance of the estimator of $m$ to decrease with $K$. For each simulation configuration we generated 20 data sets.

### 3.2 Study 1: Optimization Methods Comparison

The methods of Nelder-Mead, GA and the PSO algorithm SPSO 2011 were run with their default settings, and the same initial values of the parameters. An initial value of $m = 4$ was selected when the true $m$ was 2 or 8, and an initial $m = 6$ was chosen when the true $m$ was 4. The GA and PSO methods also allow the user to limit the range of $m$ values to search; the search limits we chose are shown in Table 3.1.

<table>
<thead>
<tr>
<th>True m</th>
<th>Initial m</th>
<th>Method</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>Nelder-Mead</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>$m \in {0, 10}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSO</td>
<td>$m \in {0, 10}$</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>Nelder-Mead</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>$m \in {0, 10}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSO</td>
<td>$m \in {0, 10}$</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>Nelder-Mead</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>$m \in {0, 15}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSO</td>
<td>$m \in {0, 15}$</td>
</tr>
</tbody>
</table>

Table 3.1: Study 1, simulation setting

The results are shown in Figure 3.1 and Table 3.2. As an indication of performance, the shaded area in each panel is the range $(0.5m, 1.5m)$. For $m = 2$, there is no obvious difference in performance between Nelder-Mead and PSO, while GA tends to overestimate. For $m = 4$, all three methods provide reasonable estimates of $m$, though Nelder-Mead always underestimates the true value. Under the largest value $m = 8$ the estimates from Nelder-Mead appear to be substantially downwardly biased, while the estimates from PSO are highly variable. Overall, GA outperforms the other two methods in terms of accuracy.

Note that there are datasets for which all three methods give similar estimates that are far below the true value. We speculate that for these datasets the simulated $\beta_k$ values are highly dispersed and are more compatible with a small value of $m$.

### 3.3 Study 2: Dimension of Genetic Variants

Though GA was the best method in Study 1, we used PSO in this study because of its reduced computation time. Following Clerc [3] we chose the number of particles in the swarm to be 25, 35, 45 and 45 for $K = 10, 20, 30$ and 50, respectively. For $K = 70$ there is no recommendation and we chose a swarm of 80 particles. The initial value of $m$ and
ranges of $m$ values to search were 6 and $(0,10)$, as in Study 1 (see Table 3.1). The frequency of estimates being between 2 and 6, which is the range of $(0.5m, 1.5m)$, along with the mean and variance of estimates of $m$ are shown in Table 3.3. Though there is a slight trend toward lower bias and variance as $K$ increases, these trends are weaker than expected. Larger numbers of simulation replicates may be needed to see the expected trends.

<table>
<thead>
<tr>
<th>$p$</th>
<th>Frequency</th>
<th>Mean</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>6.007</td>
<td>1.267608</td>
</tr>
<tr>
<td>20</td>
<td>18</td>
<td>3.576</td>
<td>0.6640005</td>
</tr>
<tr>
<td>30</td>
<td>18</td>
<td>4.794</td>
<td>0.581193</td>
</tr>
<tr>
<td>50</td>
<td>20</td>
<td>4.707</td>
<td>0.4509085</td>
</tr>
<tr>
<td>70</td>
<td>20</td>
<td>4.328</td>
<td>0.3870284</td>
</tr>
</tbody>
</table>

Table 3.3: Study 2, Frequency of acceptable estimation, mean and variance of estimation of $m$
Figure 3.1: Study 1, estimation of $m$ for settings of true $m = 2, 4, 8$ separately
Chapter 4

Real Data Modelling

4.1 Data Description

The Alzheimer’s Disease Neuroimaging Initiative (ADNI) is a longitudinal multicenter study, aiming to identify significant genetic variants for the early detection and tracking of AD [1]. We apply our methodology to a dataset obtained from the first phase of the ADNI study (abbreviated ADNI-1). ADNI-1 is a five-year study with an initial participant pool of 800 subjects, among which 200 are cognitively normal individuals (CN), 200 are diagnosed to have Alzheimer’s Disease (AD) and 400 had mild cognitive impairment (MCI). More information about the ADNI-1 study design are available on the ADNI website.

Our analysis is of the AD (case) and CN (control) subjects from ADNI-1. In particular, we use a subset of 179 CN and 144 AD subjects from a dataset prepared for Greenlaw et al. [8]. The SNPs for this study are from the top 40 candidate genes for AD in the AlzGene database on June 10, 2010 [8]. After imputation and quality control there were 490 SNPs in 33 genes. Of these we focus on the 426 SNPs in the 12 genes with 10 or more SNPs (Table 4.1). Covariates for each SNP genotype are coded as 0, 1 or 2 copies of the minor allele.

4.2 Estimating Shrinkage Parameter $m$

An assumption of our method is that covariates are independent. To assess the plausibility of this assumption we generated LDheatmaps [23] for the genes in Table 4.1 (see Appendix B). LDheatmaps display measures of linkage disequilibrium (LD) between pairs of SNPs. These graphs suggest relatively low correlation between SNPs. Under the assumption of independent covariates, and using PSO as the optimization method we estimated $m$ for each gene. To mitigate the possibility of the optimizer finding local maxima we re-ran the optimization procedure 10 times with different initial values, and took the estimate $\hat{m}$ to be the argument of the maximum. The results are shown in Table 4.2. We included some notes
Table 4.1: Genes included from ADNI-1 study

indicating that results reached the upper boundary and are regarded as unstable so that we cannot make appropriate inference about \( m \), or that the estimated \( m \) is larger than 10, which makes the log-F prior practically Normal \([7]\). The results suggest \( m \)-values of about 4 for ECE1, 5 for NEDD9, 8.5 for SORCS1 and values of 10 or more for the other genes.

\[
\begin{array}{cccc}
\text{Gene} & \hat{m} \\
\hline
\hat{m} & CR1 & >10 \\
ECE1 & 4.17 \\
\text{MTHFR} & \text{Upper bound reached} \\
\text{BIN1} & 7.76 \\
\text{NEDD9} & 4.80 \\
\text{DAPK1} & >10 \\
\text{IL33} & \text{Upper bound reached} \\
\text{SORCS1} & 8.54 \\
\text{GAB2} & \text{Upper bound reached} \\
\text{PICALM} & >10 \\
\text{SORL1} & >10 \\
\text{ADAM10} & >10 \\
\end{array}
\]

Table 4.2: Estimated \( m \) for each gene
Chapter 5

Discussion

Our method of estimating the shrinkage parameter $m$ is based on an approximated marginal likelihood. We started from the penalised profile likelihood of the log-OR parameters $\beta$ with a log-F prior, which can be regarded as a modification of Zhang’s approach. By approximating the marginal likelihood obtained by integrating out the $\beta$’s, we can estimate the shrinkage parameter. We argued that the Laplace approximation used to approximate the marginal likelihood is reasonably good for our unimodal penalized likelihood. The advantage of our approach over the Monte Carlo EM approach of Yu [27] is the reduced computation time. We explored some of the properties of the estimator by simulation and applied it to data from a genetic association study of Alzheimer’s disease.

The possible methods that could be used to maximize the approximated marginal likelihood are not limited to the three methods mentioned in this project. The factors that make optimization problems more difficult are the dimension of the parameter space and flatness or irregularity of the criterion function. In our problem the dimension is controlled by the number of covariates $K$. We expect that the search for $m$ will actually become easier as $K$ increases and we gain more replicates from the prior. We therefore focused our investigation of the optimization methods on the impact of the true $m$ on the criterion function. Our tentative conclusions from our simulation study is that the optimization problem becomes more challenging as $m$ increases. For small $m$ the Nelder-Mead and PSO approaches worked well, but for large $m$ we benefit from the stability of the GA approach. The drawback of GA is its increased computation time. One suggestion is to fit regressions under a trial value of $m$ to get a sense of the variability in the $\beta$ coefficients. If these appear concentrated about 0, suggesting a large $m$, then one might adopt GA to estimate $m$.

We applied our method to data from the ADNI-1 study. Assuming independence of SNPs within a gene we obtained estimates of $m$ in the ECE1, BIN1, NEDD9 and SORCS1 genes. These values could be used in subsequent SNP-specific logistic regression modeling. For the
other genes we studied we were not able to obtain trustworthy estimates of \( m \) and would recommend a large value of \( m=10 \), or a Gaussian prior. The genetic covariates in the ADNI-1 study were 0, 1, 2 counts, which differed from the data generation in our simulations. An area of future work is to conduct simulations with sparse count covariates.

A shortcoming of this project is that our method is not appropriate for low-dimensional datasets. Information about the shrinkage parameter \( m \) comes from multiple realizations from the prior distribution and we therefore need multiple covariates. By contrast, very high-dimensional datasets pose computation problems and may lead to poor performance of the optimization methods. For example, Helwig and Wanka [9] showed that the initialization and bound handling mechanism of particle swarm optimization can cause particles to become trapped at local maxima in high-dimensional search spaces.

The major limitation of this work is that it does not provide confidence intervals for our estimates of the shrinkage parameter \( m \). One possible approach is to obtain confidence intervals by inverting a profile likelihood ratio test. The profile likelihood is obtained as follows. For fixed \( m \) we consider the marginal log-likelihood to be a function of the \( \alpha^* \)'s. We can use a derivative-free optimization method to maximize this function over the \( \alpha^* \)'s to obtain an approximate profile likelihood value at \( m \). Repeating this procedure for a grid of \( m \) values gives an approximate profile log-likelihood for \( m \). A profile likelihood ratio test of a specific value \( m_0 \) versus \( \hat{m} \) is less than about 4. This reasoning leads to the so-called “drop-down-two” confidence interval comprised of all \( m_0 \) such that the estimated profile log-likelihood at \( m_0 \) is within about 2 of the estimated profile log-likelihood at \( \hat{m} \). Investigation of the properties of such an approach can be included in the future work.

Ultimately, the purpose of estimating \( m \) is to use it as the smoothing parameter in single-SNP logistic regression analyses. It is therefore of interest to explore the statistical properties of the log-OR estimator from the two-step process of first estimating \( m \) and then estimating log-ORs under a log-F(\( m,m \)) penalty. In addition to considering the approximate maximum likelihood estimator \( \hat{m} \), we might also use the \( m \) value at, say, the upper or lower limits of the confidence interval for \( m \). These explorations are also future work.
Bibliography


Appendix A

Implementation of Laplace Approximation

Recall the marginal likelihood for \((\alpha^*, m)\) and denote the product \(L(\alpha^*_k, \beta_k|m)\) by \(L_p\), which can be regarded as an unnormalized posterior density. Note that \(L_p\) is differentiable, and denote the maxima of \(L_p\) with \(\alpha^*_k, m\) given, as \(\beta_k^{\text{max}}\). \(\int L_p \, d\beta_k\) can be approximated with

\[
L_p|_{\beta_k^{\text{max}}} \sqrt{\frac{2\pi}{c P}}, c_P = -\frac{\partial^2}{\partial \beta_k^2} \log(L_p)|_{\beta_k^{\text{max}}}
\]  

(A.1)

In Practice, the value of \(L_p\) can be too small to compute in R, instead we computed \(\log(L_p)\) to access the value of \(\beta_k^{\text{max}}\) by simply taking the derivatives. Plug-in (2.4) and (2.5) we have

\[
\log(L_p) = \sum_{i=1}^{n} (Y_i(\alpha_k^* + X_i \beta_k) - \log(1 + \exp(\alpha_k^* + X_i \beta_k)))
\]

(A.2)

\[
-\log(Beta(m/2, m/2)) - \frac{m}{2} \beta_k - m * \log(1 + \exp(-\beta_k))
\]

\[
\frac{\partial \log(L_p)}{\partial \beta_k} = \sum_{i=1}^{n} (Y_iX_i^k - \frac{\exp(\alpha_k^* + X_i \beta_k)X_i}{1 + \exp(\alpha_k^* + X_i \beta_k)}) - \frac{m}{2} + m \frac{\exp(-\beta_k)}{1 + \exp(-\beta_k)}
\]  

(A.3)

To show that \(L_p\) is well-peaked enough for Laplace approximation, we prove the following result to ensure its unimodality:

**Result** The root of \(\partial \log(L_p)/\partial \beta_k = 0\), denoted by \(\beta_k^{\text{max}}\), is the global maxima of \(L_p\).

**Proof** Rewrite (A.3) with notations of \(e_\alpha = \exp(\alpha_k^*), e_k = \exp(\beta_k)\), we have

\[
\sum_{i=1}^{n} (Y_iX_i^k - \frac{e_\alpha e_k X_i^k}{1 + e_\alpha e_k}) - \frac{m}{2} + m \frac{1/e_k}{1 + 1/e_k} = \sum_{i=1}^{n} (Y_iX_i^k - X_i^k + \frac{X_i^k}{1 + e_\alpha e_k}) - \frac{m}{2} + \frac{m}{1 + e_k}
\]  

(A.4)
Consider when $\beta_k \to -\infty, e_k \to 0, \frac{X_i^k}{1+e_a e_k^i} \to X_i^k$ when $X_i^k > 0; \to 0$ when $X_i^k < 0$. Then

$$\sum_{i=1}^{n} (Y_i X_i^k - X_i^k) + \frac{X_i^k}{1+e_a e_k^i} - m + \frac{m}{2} + \frac{m}{1+e_k}$$

$$= \sum_{i=1}^{n} (Y_i X_i^k - X_i^k I(X_i^k > 0) + \frac{m}{2}$$

Similarly, when $\beta_k \to \infty, e_k \to \infty, \frac{X_i^k}{1+e_a e_k^i} \to 0$ when $X_i^k > 0; \to X_i^k$ when $X_i^k < 0$. We have

$$\sum_{i=1}^{n} (Y_i X_i^k - X_i^k) + \frac{X_i^k}{1+e_a e_k^i} - m + \frac{m}{2} + \frac{m}{1+e_k}$$

$$= \sum_{i=1}^{n} (X_i^k I(Y_i = 1) I(X_i^k < 0) - X_i^k (Y_i = 0) I(X_i^k > 0)) - \frac{m}{2} < 0$$

Since (A.3) is continuous on $\mathbb{R}$, a root of $\partial \log(L_p)/\partial \beta_k = 0$ must exist according to intermediate value theorem. Next we prove this root, denoted by $\beta_k^{\text{max}}$ is the only root. The Hessian

$$\frac{\partial^2}{\partial \beta_k^2} \log(L_p) = \sum_{i=1}^{n} \left( -\frac{\exp(a_i^* + X_i^k \beta_k) (X_i^k)^2}{1 + \exp(a_i^* + X_i^k \beta_k)} + \frac{(\exp(a_i^* + X_i^k \beta_k))^2 (X_i^k)^2}{(1 + \exp(a_i^* + X_i^k \beta_k))^2} \right)$$

$$+ m\left( \frac{-\exp(-\beta_k)}{1 + \exp(-\beta_k)} + \frac{(\exp(-\beta_k))^2}{(1 + \exp(-\beta_k))^2} \right)$$

$$= \sum_{i=1}^{n} \left( (-1 + \frac{\exp(a_i^* + X_i^k \beta_k)}{1 + \exp(a_i^* + X_i^k \beta_k)}) \frac{\exp(a_i^* + X_i^k \beta_k) (X_i^k)^2}{1 + \exp(a_i^* + X_i^k \beta_k)} \right)$$

$$+ m\left( (-1 + \frac{\exp(-\beta_k)}{1 + \exp(-\beta_k)}) \frac{\exp(-\beta_k)}{1 + \exp(-\beta_k)} \right)$$

is always $< 0$ for all $\beta_k \in \mathbb{R}$, since the function $\frac{\exp(u)}{1 + \exp(u)}$ is always $< 1$ for all $u \in \mathbb{R}$. This gives that (A.2) is a monotonically decreasing function on $\mathbb{R}$, and $\beta_k^{\text{max}}$ is the only root of $\partial \log(L_p)/\partial \beta_k = 0$. Finally, $\partial \log(L_p)/\partial \beta_k > 0$ when $-\infty < \beta_k < \beta_k^{\text{max}}$; $< 0$ when $\beta_k^{\text{max}} < \beta_k < \infty$, which indicates that $\beta_k^{\text{max}}$ is the global maxima of $\log(L_p)$, and $L_p$. □
Appendix B

LDheatmaps for Genes in Real Data Analysis

Here we present the LDheatmaps for genes included in our real data analysis of ADNI-1 study. There were indeed SNPs with high pairwise correlation, especially for genes with fewer SNPs. We could have removed part of the SNPs to approach the independence of covariates on one hand, and on the other hand, this would reduce the dimension of covariates and information of $m$.

The estimated $m$ for ECE1 and NEDD9 in our results from the full set of available SNPs seem to be more plausible based on their LDheatmaps - the pairwise correlation is generally low except for SNPs that are close to each other.
Figure B.1: LDheatmaps for genes included in real data analysis section, using $R^2$ measure of LD (part 1)
Figure B.2: LDheatmaps for genes included in real data analysis section, using $R^2$ measure of LD(part 2)
Appendix C

Code

```r
library(pso)
library(GA)

n<-1000
mo<-2  # the prior density of beta is log-F
p<-19

# the prior density of beta is log-F

simUnmatched = function(n,p,scale=FALSE){
  n is total sample size, beta1 is value of parameter of interest,
  # p is number of nuisance covariates
  ConCaseRatio = 4  # assuming 4:1 con:case ratio

  ncase = n/(ConCaseRatio+1); ncon=ncase*ConCaseRatio
  beta = log(rf(p+1,mo/2,mo/2))  # p nuisance params of value 1
  ncov = p+1

  # Simulate cases and controls
  conX = caseX = NULL
  for(i in 1:ncov) {
    conX = cbind(conX,rnorm(ncon,mean=0,sd=1))
    caseX = cbind(caseX,rnorm(ncase,mean=beta[i],sd=1))
  }
  X = rbind(caseX,conX)
  colnames(X) = paste0("x",1:ncov); rownames(X) = NULL
  case = c(rep(1,ncase),rep(0,ncon))
  return(data.frame(case,X))
}

# Simulate cases and controls

nrep<-20
irep<-1
est.m<-numeric(nrep)
est.m.NM<-numeric(nrep)
est.m.GA<-numeric(nrep)

psoLA<-function(alpha_star.V,m,n.rounds){
  tracer<-matrix(0,nrow=1,ncol=p+4)
  ftracer=0
  for (i in 1:n.rounds){
    beta_max<-numeric(p+1)
    for (di in 1:(p+1)){
      X<-X-M[,di]
      alpha_star<-alpha_star.V[di]
      dlogPenalisedL<-function(beta){
        sum(X*y-(X*exp(alpha_star+beta*as.numeric(X)))/(1+exp(alpha_star+beta*as.numeric(X))))
        -m/2+m*exp(-beta)/(1+exp(-beta))
      }
      beta_max[di]<-uniroot(dlogPenalisedL, c(-20,20))$root
    }
    multi.dimen.logLP_betamax<-function(alpha_star0m0){
      m0=alpha_star0m0[p+2]
      ll<-0
      for (di in 1:(p+1)){
```
alpha_star0 <- alpha_star0m0[di]
X <- XM[,di]
temp1 <- -sum(X^2 * exp(alpha_star0 + beta_max[di]*X) / (1 + exp(alpha_star0 + beta_max[di]*X)))
- sum(X^2 * (exp(alpha_star0 + beta_max[di]*X) / (1 + exp(alpha_star0 + beta_max[di]*X)))^2)
temp2 <- -(exp(-beta_max[di]) / (1 + exp(-beta_max[di])))^2
c <- temp1 + m0 * temp2
LP_di <- -sum(y*(alpha_star0 + beta_max[di]*as.numeric(X)) - log(1 + exp(alpha_star0 + beta_max[di]*as.numeric(X))))
-log(beta(m0/2, m0/2)) - m0/2 * beta_max[di] - m0 * log(1 + exp(-beta_max[di])) - 0.5 * log(c)
ll <- ll + LP_di
}
ll
pso.result <- psoptim(par = c(alpha_star.V, m), fn = multi.dimen.logLP_betamax, lower = c(rep(-20, p + 1), 0), upper = c(rep(10, p + 1), 15), control = list(trace = 100, fnscale = -1, maxit = 3000, maxit.stagnate = 50, s = 50, type = "SPSO2011"))
if (abs(pso.result$value - ftracer) >= 0.001 * abs(ftracer)) {
alpha_star.V <- pso.result$par[1:(p + 1)]
m <- pso.result$par[p + 2]
tracer <- rbind(tracer, c(as.integer(i), alpha_star.V, m, pso.result$value))
ftracer <- pso.result$value
print(tracer[i + 1,])
} else {
break
}
}
tracer

NMLA <- function(alpha_star.V, m, n.rounds) {
tracer <- matrix(0, nrow = 1, ncol = p + 4)
ftracer <- 0
i = 1
for (i in 1:n.rounds) {
beta_max <- numeric(p + 1)
for (di in 1:(p + 1)) {
alpha_star <- alpha_star.V[di]

dlogPenalisedL <- function(beta){
sum(X*y - (X*exp(alpha_star + beta*as.numeric(X)) / (1 + exp(alpha_star + beta*as.numeric(X)))) - m/2 + m*exp(-beta) / (1 + exp(-beta))
}
beta_max[di] <- uniroot(dlogPenalisedL, c(-20, 20))$root
}
multi.dimen.logLP_betamax <- function(alpha_star0m0) {
m0 = alpha_star0m0[p + 2]
l1 <- 0
for (di in 1:(p + 1)) {
alpha_star0 <- alpha_star0m0[di]
X <- XM[,di]
temp1 <- -sum(X^2 * exp(alpha_star0 + beta_max[di]*X) / (1 + exp(alpha_star0 + beta_max[di]*X)))
- sum(X^2 * (exp(alpha_star0 + beta_max[di]*X) / (1 + exp(alpha_star0 + beta_max[di]*X)))^2)
temp2 <- -(exp(-beta_max[di]) / (1 + exp(-beta_max[di])))^2
c <- temp1 + m0 * temp2
LP_di <- -sum(y*(alpha_star0 + beta_max[di]*as.numeric(X)) - log(1 + exp(alpha_star0 + beta_max[di]*as.numeric(X))))
-log(beta(m0/2, m0/2)) - m0/2 * beta_max[di] - m0 * log(1 + exp(-beta_max[di])) - 0.5 * log(c)
ll <- ll + LP_di
}
ll
opt.result <- optim(par = c(alpha_star.V, m), fn = multi.dimen.logLP_betamax, method = "Nelder-Mead", control = list(fnscale = -1))
if (abs(opt.result$value - ftracer) >= 0.001 * abs(ftracer)) {
alpha_star.V <- opt.result$par[1:(p + 1)]
m <- opt.result$par[p + 2]
tracer <- rbind(tracer, c(as.integer(i), alpha_star.V, m, opt.result$value))
ftracer <- opt.result$value
print(tracer[i + 1,])
} else {
break
}
GALA <- function(alpha_star.V, m, n.rounds) {
  tracer <- matrix(0, nrow = 1, ncol = p + 4)
  ftracer <- 0
  i = 1
  for (i in 1:n.rounds) {
    beta_max <- numeric(p + 1)
    for (di in 1:(p + 1)) {
      X <- XM[, di]
      alpha_star <- alpha_star.V[di]
      dlogPenalisedL <- function(beta) {
        sum(X * y - (X * exp(alpha_star + beta * as.numeric(X))) / (1 + exp(alpha_star + beta * as.numeric(X))))
        - m / 2 + m * exp(-beta) / (1 + exp(-beta))
      }
      beta_max[di] <- uniroot(dlogPenalisedL, c(-20, 20))$root
    }
    mult.dimen.logLP_betamax <- function(alpha_star0m0) {
      m0 = alpha_star0m0[p + 2]
      ll <- 0
      for (di in 1:(p + 1)) {
        alpha_star0 <- alpha_star0m0[di]
        X <- XM[, di]
        temp1 <- sum(X^2 * exp(alpha_star0 + beta_max[di] * X) / (1 + exp(alpha_star0 + beta_max[di] * X)))
        temp2 <- exp(-beta_max[di]) / (1 + exp(-beta_max[di]))^2
        c = temp1 + m0 * temp2
        LP_di <- sum(y * (alpha_star0 + beta_max[di] * as.numeric(X)) - log(1 + exp(alpha_star0 + beta_max[di] * as.numeric(X))))
        - m0 / 2 * beta_max[di] - m0 * log(1 + exp(-beta_max[di])) - 0.5 * log(c)
        ll <- ll + LP_di
      }
      ll
    }
    GA <- ga(type = "real-valued", fitness = function(x) -(-mult.dimen.logLP_betamax(x)),
             lower = c(rep(-20, p + 1), 0), upper = c(rep(10, p + 1), 15),
             popSize = 50, maxiter = 2000,
             run = 50, parallel = 4)
    if (abs(GA@fitnessValue - ftracer) >= 0.001 * abs(ftracer)) {
      alpha_star.V <- GA@solution[1:(p + 1)]
      m <- GA@solution[p + 2]
      tracer <- rbind(tracer, c(as.integer(i), alpha_star.V, m, GA@fitnessValue))
      ftracer <- GA@fitnessValue
      print(tracer[i + 1, ])
    } else {
      break
    }
  }
}

for (irep in 1:nrep) {
  simdata <- simUnmatched(n, p)
  y <- simdata$case
  XM <- as.matrix(simdata[, -1])
  # glm <- glm(y ~ XM, family = binomial)
  # Initialization
  ini.alpha <- rep(unname(-5), p + 1)
  ini.m <- 4
  n_rounds <- 400
  tracer1 <- psoLA(ini.alpha, ini.m)
  tracer2 <- NMLA(ini.alpha, ini.m)
  tracer3 <- GALA(ini.alpha, ini.m)
  est.m[irep] <- tracer1[nrow(tracer1), p + 3]
  est.m.NM[irep] <- tracer2[nrow(tracer2), p + 3]
  est.m.GA[irep] <- tracer3[nrow(tracer3), p + 3]
  irep <- irep + 1
}