Università degli Studi di Padova
Dipartimento di Scienze Statistiche
Corso di Laurea Magistrale in
Scienze Statistiche

Item-Response Model Estimation Via Penalized Likelihood

Relatore Prof. Nicola Sartori
Dipartimento di Scienze Statistiche

Laureando: Bruno Zamengo
Matricola 1067767

Anno Accademico 2015/2016

Contents

List of Figures

Introduction

Nowadays people have to attend tests very frequently. The tests are used to evaluate their ability or to understand their knowledge about a particular argument. To achieve such result we need to know the difficulty of each problem of the test so we need to evaluate people ability and question difficulty together. Georg Rasch, a Danish statistician, has been the first person who attempted to model the probability a subject makes a certain amount of errors in a test made up by a fixed number of questions, also called items (Fienberg 2004). The model proposed by Rasch belongs to the family of models of the *Item-Response Theory (IRT)* and, within this family it is called 1PL model. IRT models fit the probability a subject, say the sth, answers correctly an item, say the ith, with a parameter which depends on both the subject and the item indexes. Let us call such parameter π_{si}. Rasch proposed to model π_{si} with a logistic regression which uses one parameter per subject and one parameter per item. The subject parameters measure the ability of the subjects while the item parameters measure the difficulty of the items. The latter are relative measures, that is the item parameters measure how more or how less difficult an item is compared to an item used as reference. The Rasch model, also called 1PL model, represents the first step among the class of Item-Response models. One of its extensions is the 2PL model that adds one parameter per item which is interpreted as its discriminating power, that is how much that item discriminates the subject ability.
The 1PL model belongs to the class of generalized linear models so its estimation can be worked out without problems. The 2PL model however is non linear in the parameter set and its estimation is troublesome. This thesis proposes to estimate such models first assuming random effects for the subject parameters , which is the standard in literature, and then using some penalization on the likelihood function for the discrimination parameters.

This thesis uses the paradigm of the likelihood theory which belongs to the so-called frequency-decision paradigm. Such paradigm and the likelihood theory are briefly reviewed in Chapter 1.

Chapter 2 gives a review on generalized linear model in general and the logistic regression in particular because, as mentioned above, the Rasch

model belongs to the class of generalized linear models and more specifically it is a logistic regression, and the 2PL model is somehow connected with them. This chapter offers an overview of generalized linear mixed models as well because they can be used to threat random effects which we use when dealing with the 2PL model.

Because this thesis purpose is trying to estimate the 2PL model with some penalization of the likelihood function, Chapter 3 deals with the most used penalization techniques such as the LASSO and ridge regressions and L_q and elastic net penalties, even though only the LASSO will be used.

The Item-Response Theory itself is presented in Chapter 4, in particular it analyses the 1PL and 2PL model likelihood functions. We present here our proposal to handle the 2PL model, that is we assume random effects for the subject parameters and then we penalize the resulting likelihood function in the hope we get a model in between 1PL and 2PL models.

Chapter 5 presents the most operative part of this document. Here we present the simulation procedure we have followed and we analyse the collected results. Our purpose is understanding if the proposal of penalizing the marginal likelihood is useful to regularize the model and if it helps identifying which parameters are really present in the model. We also compare different methods to select the tuning parameter of the model.

This document has an appendix which reports the used code to allow readers to reproduce the simulations.

1 | Likelihood Theory

This thesis follows the so-called frequency-decision paradigm. This introductory chapter purpose is briefly recalling the paradigm basic concepts such as the definition of likelihood function.

1.1 Introduction

The purpose of statistical inference is to summarize the observed data y by reconstructing its true probability distribution function. This process can be split roughly into three different steps:

1. Model specification;
2. Statistical inference;
3. Empirical model validation.

The first step will be reviewed in Paragraph 1.2 while the other paragraphs of this chapter will deal with the second one. Model validation is an essential part of any statistical analysis but it is not dealt in this thesis.

1.2 Model specification

A statistical model $\mathcal{F}$ is a probability distribution family where each of its members is, at least qualitatively, compatible with the observed data y. When $p_0(\cdot)$, the data true probability distribution, belongs to $\mathcal{F}$ then $\mathcal{F}$ is said *correctly specified*. If $p_0(\cdot) \notin \mathcal{F}$ then $\mathcal{F}$ is said *misspecified*.

On real data studies, statistical models are considered approximate description of the true probability distribution which captures the features studied. Depending on the level of specification, the statistical model can belong to any of the following classes:

Parametric Model: this model specification is used when previous knowledge of the phenomenon is available. $\mathcal{F}$ is a restricted class of distributions indexed by a finite-dimensional parameter, that is

$$\mathcal{F} = \{p(y, \theta), \quad \theta \in \Theta \subseteq \mathbb{R}^p\}.$$

This kind of models are called parametric because the difference between two distribution lies in the difference of the θ parameter. The true model is $p_0(y) = p(y, \theta_0)$ and θ_0 is the true parameter value. A correct specification is determined by the relation $\theta_0 \in \Theta$: if the relation is verified then the model is correctly specified otherwise the problem is misspecified.

Non-parametric Model: when the rules which discriminate the probability distributions do not identify a finite-dimension parameter but they just give a set of simplifying assumption like the random variable support or the distribution function mathematical properties, the model is called non-parametric.

Semi-parametric Model: this kind of specification is a mix of the previous ones. The elements of $\mathcal{F}$ are identified by both a parameter and some simplifying assumption. Formally it can be represented as

$$\mathcal{F} = \{p(y, \theta), \quad \theta \in \Theta\},$$
$$\text{where } \theta = (\tau, h(\cdot)),$$

with τ a finite-dimension parameter, whereas $h(\cdot)$ cannot be index by a finite-dimension parameter. The simplifying assumptions are referred to $h(\cdot)$.

The three model macro-categories can be sorted by their specification level: the parametric model requires the deepest specification level but it grants small-sized distribution families whereas the non-parametric level requires just a few specifications but it defines wide distributions families. The semi-parametric is somehow a compromise of the previous two.

1.3 The likelihood function

This work is related to parametric models so the purpose is to identify the most likely parameter set given the observed data. The identification of the model parameters from the observed data is called estimation process. The identified parameter set is called estimate.

A popular estimation procedure is given by the likelihood theory, developed by Fisher in 1922 (Fisher 1922). It is based upon the likelihood function and, via its maximization, it offers a standard procedure to estimate the unknown parameters.

Fisher's likelihood definition is the following (Fisher 1922):

> The likelihood that any parameter (or set of parameters) should have any assigned value (or set of values) is proportional to the probability that if this were so, the totally of observations should be that observed.

This means, given a set of observed data y, the likelihood function computed on the θ vector is proportional to the probability of observing the y set if the true parameter value is θ,

$$\mathcal{L}(\theta, y) = c(y)p(y, \theta),$$

where $c(y)$ is a positive factor which depends on the data only. Often the likelihood function is referred as $\mathcal{L}(\theta)$, omitting the y data underlining since the likelihood function is computed on the model parameter θ with y fixed at the observed data. It is very important to say the multiplicative factor $c(y)$ must be non negative.

The likelihood theory owes its popularity to its simplicity and generality, in fact it provides general approximations of the sampling distributions of the estimated quantities. These approximations are based on some probability limit theorems which are valid when n, the sample size, diverges, i.e. $n \to \infty$. The sample size index n, in likelihood theory, should be considered as the amount of information brought by the available data: as it increases, the inference procedure should get more precise.

Likelihood asymptotic results are valid when some regularity hypotheses are met. When a parametric problem respects all of them it is called *regular (parametric) model* (Azzalini 1996). Regularity assumptions are referred to the statistical model $\mathcal{F} = \{p(y, \theta), \theta \in \Theta \subseteq \mathbb{R}^p\}$ and they are:

1. the model must be identifiable that means a one-to-one correspondence between Θ elements and $\mathcal{F}$ elements exists;

2. the support of Y must not depend on θ and it is common to all the elements of $\mathcal{F}$;

3. the parametric space $\Theta \subseteq \mathbb{R}^p$ is an open set and p is not influenced by n, the sample size;

4. the likelihood function is at least three-times differentiable (in θ) with continuous derivatives.

1.4 Likelihood related quantities

From the likelihood definition it comes naturally that an estimate of θ, $\hat{\theta}$, is the value of the parameter which maximizes the likelihood function. When dealing with regular problems, the parameter set which maximizes the likelihood function can be worked out with a standard procedure. This procedure involves some quantities related to the likelihood function. This section will present them.

1.4.1 Log-likelihood function

The likelihood function computed for a parameter set θ is defined as the probability of observing the y data if the underlying model is indexed by θ. Usually y is made up by many observations which could make the computation of $p(y, \theta)$ hard, very often models make the hypothesis that the data are independent random variables Y_i. This assumption makes the analytical expression of $p(y, \theta)$ far easier, i.e.

$$p(y, \theta) = \prod_{i=1}^{n} p_{Y_i}(y_i, \theta)$$

where n represents the sample size, that is the size of the observed data and y_i is the ith observation. When Y_i $i = 1, \dots n$ have all the same distribution there is additional simplification.

Under the independence assumption, the likelihood function is the product of n terms. Maximizing a function often involves the derivative and when the function is the product of n terms this can be gruelling and computationally onerous. To simplify the problem it is convenient considering a transformation of the likelihood function, its natural logarithm, which is a monotone transformation. Such function is called *log-likelihood function* and it is denoted by

$$\ell(\theta) = \log \mathcal{L}(\theta).$$

Under the assumption of independence the log-likelihood function can be expressed as

$$\ell(\theta) = \sum_{i=1}^{n} \log p_{Y_i}(y_i, \theta),$$

which is made up by the sum of n terms. This offers some theoretical support too because the most important asymptotic probability results involve the sum of random quantities.

1.4.2 Score function

The score function is defined as the derivative function of the log-likelihood, that is

$$\ell_*(\theta) = \frac{\partial}{\partial \theta} \ell(\theta).$$

If the parameter is p dimensional the score function is the log-likelihood gradient. The score function can be computed if and only if the likelihood function is differentiable at least once. When the model is regular the score function can always be computed.

Due to its definition, the score function is used to find the estimate of θ via the equation $\ell_*(\theta) = 0$ which is called likelihood equation.

1.4.3 Observed and expected information

Under regularity assumptions, the likelihood function is at least three times differentiable so the second order derivative function of the log-likelihood can be used with the score function to find out which is the parameter set that maximizes the likelihood function. The second order derivative function is usually referred to as

$$\ell_{**}(\theta) = \frac{\partial^2}{\partial\theta\partial\theta^\intercal}\,\ell(\theta) = \frac{\partial}{\partial\theta^\intercal}\,\ell_*(\theta).$$

When the parameter θ is p dimensional the score function is the log-likelihood Hessian. The opposite of the log-likelihood Hessian computed in θ is called observed information and it is usually denoted by $j(\theta) = -\ell_{**}(\theta)$. The expected value of the observed information is called expected information and it is denoted by $i(\theta) = \mathrm{E}[j(\theta)]$.

1.4.4 Maximum Likelihood Estimate and Estimator

The $\hat{\theta}$ value which maximizes the likelihood function is called *maximum likelihood estimate*. Usually this value is found by solving the likelihood equation $\ell_*(\theta) = 0$.

Under the principle of repeated sampling, the likelihood equation, before observing the data, is a random quantity. The solution of such equation before the data are observed is called *maximum likelihood estimator* and it is usually referred to as $\hat{\theta}_n$ or $\hat{\theta}_n(Y)$ to denote it as a function of the data when they will be observed. Because the maximum likelihood estimator is computed before observing the data, it is a random quantity itself.

1.5 Likelihood asymptotic results

The likelihood theory owns its wide use to its generality and its asymptotic results. While the former has been reviewed in the previous paragraphs, the latter will be briefly presented here. The following results hold when the regularity assumptions are met. This paragraph reports only the main results and does not provide their proofs. Further details can be found, for instance, in Pace and Salvan (1997).

Maybe one of the most important result is the maximum likelihood estimate consistency: when the sample size diverges the estimate converges in probability and, under slightly stronger conditions, almost surely to the true parameter value θ_0. This means the maximum likelihood estimate is a good choice because, as the amount of available information grows, the fitted estimate "falls" nearer to the parameter true value.

The maximum likelihood estimate consistency is related to the unbiased property of the likelihood equation. Indeed, when the model is regular,

the first Bartlett identity asserts the likelihood equation is unbiased, that is $\mathrm{E}_\theta\big[\ell_*(\theta, Y)] = 0$. There is a second Bartlett identity as well that involves the variance of the score function: when the model is regular, the variance of the score function is the expected information, that is $\mathrm{Var}_\theta[\ell_*(\theta, Y)] = i(\theta)$.

The two Bartlett identities are used to show that, before observing the data, the score function is asymptotically a normal random variable with null mean and variance the expected information, which is

$$\ell_*(\theta, Y) \xrightarrow{\mathrm{d}} \mathrm{N}_p(0, i(\theta)).$$

This result is used to test hypothesis as well as to get the asymptotic distribution of the maximum likelihood estimator: it can be proved, under the regularity conditions and with the score asymptotic result, the maximum likelihood estimator is asymptotically normal distributed with mean the parameter true value θ and variance $i^{-1}(\theta)$, that is

$$\hat{\theta}_n \xrightarrow{\mathrm{d}} \mathrm{N}_p(\theta, i^{-1}(\theta)).$$

Because the true value θ is unknown $i(\theta)$ can be approximated by either $i(\hat{\theta})$ or $j(\hat{\theta})$. Again, this result is used to test hypothesis. The statistical test based on this approximation is called Wald test. The Wald test is very popular due to its simplicity and generality but it has also some limits. For instance, it is not invariant under reparameterizations of the model.

In a neighbourhood of θ, the parameter true value, the likelihood can be approximated with a Taylor series. Considering the terms until order 2 it is

$$\ell(\theta) \doteq \ell(\hat{\theta}) + \ell_*(\hat{\theta})(\hat{\theta} - \theta) + \frac{1}{2}(\hat{\theta} - \theta)^\intercal \, \ell_{**}(\hat{\theta})(\hat{\theta} - \theta).$$

The second term is zero because $\ell_*(\hat{\theta}) = 0$ due to the definition of $\hat{\theta}$. Rearranging the terms we get the *parabolic approximation of the likelihood function*, which is

$$\ell(\theta) - \ell(\hat{\theta}) \doteq -\frac{1}{2}\,(\hat{\theta} - \theta)^\intercal j(\hat{\theta})(\hat{\theta} - \theta).$$

The parabolic approximation of the likelihood function with the asymptotic distribution of the maximum likelihood estimator are used to show the likelihood ratio asymptotic distribution is χ^2_p, that is

$$W(\theta) = 2\{\ell(\hat{\theta}) - \ell(\theta)\} \xrightarrow{\mathrm{d}} \chi^2_p.$$

Again, this result is mainly used to test hypothesis. The likelihood ratio test should be preferred to the Wald test because it has better properties and its convergence to the asymptotic distribution is often faster.

2 | Generalized Linear Models

Often the analysed data have asymmetric roles: some variables can be controlled or the interest of the analyst lies on a restricted subset. When such situation is verified the study purpose is to find a model which explains the behavior of the variables. The formers are called *explicative variables* or *regressors* while the latter are *response variables*. This chapter deals with problems where there are many regressors, say p, and just one response variable. The formers will be indexed by the letter x while the latter will be labelled with y. The whole problem can be mathematically summed up as

$$\mathrm{E}[Y] = f(x, \theta).$$

In parametric models, such as the ones described in this chapter, f is a well defined function from $\mathcal{X} \subseteq \mathbb{R}^p$ to $\mathcal{Y} \subseteq \mathbb{R}$. The study purpose is selecting the parameter $\hat{\theta}$ which best fits the available data.

2.1 Linear models

Among the family of functions from $\mathbb{R}^p$ to $\mathbb{R}$ the simplest form is the linear one. Models in the form $\mathrm{E}[Y] = x_1\beta_1 + \dots x_p\beta_p$ are called *linear models*. Linear models are linear in β, the coefficient vector, and not in x, the set of regressors. Linear models are very useful when the response variable support spreads on the whole $\mathbb{R}$ set. A more formal definition of linear models is

$$Y_i = \sum_{j=1}^{p} \beta_j x_{ij} + \varepsilon_i, \tag{2.1}$$

where ε_i is the error term. Linear models make three hypothesis on ε_i, called second order hypotheses:

1. $\mathrm{E}[\varepsilon_i] = 0 \qquad \forall\, i = 1, 2, \dots n;$
2. $\mathrm{Var}[\varepsilon_i] = \sigma^2 \qquad \forall\, i = 1, 2, \dots n;$
3. $\mathrm{cov}(\varepsilon_i, \varepsilon_j) = 0 \qquad \forall\, i \neq j;$

Often a further simplifying assumption is added: the ε variables are normally distributed. When this further assumption is verified, the asymptotic approximations coincide with the true estimators distributions.

Under the second order hypotheses, the maximum likelihood estimate for β can be expressed analytically

$$\hat{\beta} = (XX^{\intercal})^{-1}X^{\intercal}y,$$

where $X = (x_1, \dots x_p)$ is the regression matrix made up by p columns, the ith containing the set of observed values for the ith regressor, x_i. From the asymptotic results of the likelihood theory, it can be shown the $\hat{\beta}$ estimator is asymptotically normal distributed

$$\hat{\beta} \dot{\sim} \mathrm{N}_p\left(\beta, (XX^{\intercal})^{-1}\sigma^2\right).$$

When the ε_i are normal, then the above distribution is exact.

2.2 Introduction to generalized linear models

Linear models are a powerful modelling instrument but sometimes the hypothesis that the response variable support spreads the whole $\mathbb{R}$ set is not met:

- some variables, such as survival time of people or machines, are strictly positive;
- some are count-data, such as the number of clicks in a web page;
- some data are binary, like employed/unemployed;
- some data can be categorical as people educational qualification.

These kind of data can hardly be represented with a continuous real variable. To overcome this issue, a generalization of the linear models has been introduced. Due to its nature, this new model class has been named *generalized linear models*.
Consider the following definition of linear model:

$$\begin{aligned} Y_i|\underline{x}_i^{\intercal} &\sim f(\cdot), \\ \mathrm{E}[Y_i|\underline{x}_i^{\intercal}] &= \sum_{j=1}^{p} \beta_j x_{ij}, \\ \mathrm{Var}[Y_i] &= \sigma^2 \qquad \forall\, i = 1, 2, \cdot n, \\ \mathrm{cov}(Y_i, Y_j) &= 0 \qquad \forall\, i \neq j. \end{aligned} \tag{2.2}$$

Equations (2.2) are equivalent to Equation (2.1) with 1, 2 and 3 except for the error term (ε_i): this piece of information has been included in the

conditional distribution of Y_i given $\underline{x}_i^\intercal$. The generalization introduced by generalized linear models redefines the conditional expected value of Y_i and relaxes the homoschedasticity hypothesis. Generalized linear model's definition of conditional expected value is not the linear predictor itself but some transformation, that is

$$\mathrm{E}[Y_i|\underline{x}_i^\intercal] = g\left(\sum_{j=1}^{p}\beta_j x_{ij}\right).$$

The $g(\cdot)$ function is *called link* function. It can be proved that the asymptotic results obtained for linear models hold for generalized linear models coefficient estimators as well, as long as maximum likelihood estimator is considered.

2.3 Generalized linear model for binary data

The purpose of this thesis is the study of Item Response Theory (IRT) models, in particular generalization of the Rasch model (Fienberg 2004). These models deal mainly with binary data, so, among the family of generalized linear models, only models for binary data, and in particular the logistic model, will be analysed.

Binary data can be thought of as the realization of a dichotomous variable. Usually one of the two possible responses is more scientifically interesting so it's called "success" while the other one is called "failure". This scheme can be summed up with a 0/1 variable, a successes corresponds to 1 while 0 is assigned to failure. The probability of getting a success is not generally equal to the probability of getting a failure so the considered variable should be indexed by a parameter, say π, which represents the success probability. This model describes a Bernoulli random variable of index π.

When the response variable is dichotomous it can be represented with a Bernoulli variable of index π_i: the success rate depends on the subject. In this particular use the generalized linear model considers the conditional mean of the response variable with a function of the linear predictor,

$$\mathrm{E}[Y_i|\underline{x}_i^\intercal] = \pi_i = g\left(\sum_{j=1}^{p}\beta_j x_{ij}\right).$$

The log-likelihood function is

$$\begin{aligned}\ell(\beta) &= \sum_{i=1}^{n} y_i \log \pi_i + (1-y_i)\log(1-\pi_i) \\ &= \sum_{i=1}^{n} y_i \log g\left(\sum_{j=1}^{p}\beta_j x_{ij}\right) + (1-y_i)\log\left(1-g\left(\sum_{j=1}^{p}\beta_j x_{ij}\right)\right) \\ &= \sum_{i=1}^{n} y_i \operatorname{logit} g\left(\sum_{j=1}^{p}\beta_j x_{ij}\right) + \log\left(1-g\left(\sum_{j=1}^{p}\beta_j x_{ij}\right)\right),\end{aligned}$$

where $\operatorname{logit}(x) = \log\frac{x}{1-x}$. The link function $g(\cdot)$ has to map $\mathbb{R}$ to $[0,1]$. There are plenty of functions which meet this requirement but the following are the most widely used:

- inverse logistic function: $\operatorname{logit}^{-1}(x) = \frac{e^x}{1+e^x}$;
- "probit" function: $\operatorname{probit}(x) = \Phi(x)$, where $\Phi(\cdot)$ is the standard normal distribution function;
- inverse complementary log log function: $\operatorname{cloglog}(x) = \log\bigl(-\log(1-x)\bigr)$.

Among the proposed functions, the inverse logistic is the most widely used because it simplifies the log-likelihood function and allows meaningful parameter interpretation. When $g(\cdot) = \operatorname{logit}^{-1}(\cdot)$ the regression model is called *logistic regression* and the log-likelihood is

$$\ell(\beta) = \sum_{i=1}^{n} y_i \sum_{j=1}^{p}\beta_j x_{ij} - \log\left(1 - e^{\sum_{j=1}^{p}\beta_j x_{ij}}\right).$$

For compactness it is good practice to rewrite the log-likelihood function with matrix notation

$$\ell(\beta) = \sum_{i=1}^{n} y_i \underline{x}_i^{\mathsf{T}}\beta - \log\left(1 - e^{\underline{x}_i^{\mathsf{T}}\beta}\right),$$

where $\underline{x}_i^{\mathsf{T}}$ is the ith row of the regression matrix. Often the linear predictor, $\underline{x}_i^{\mathsf{T}}\beta$, is referred to as η_i.

While the linear model $\hat{\beta}_n$ estimator can be expressed analytically, in the logistic model and, more generally, generalized linear models, estimators of $\hat{\beta}_n$ cannot be expressed analytically but numeric iterative algorithms have been developed. In particular a *Fisher-scoring method* is used. A description can be found, for instance, in Pace and Salvan (1997, page 237).

2.4 Generalized Linear Mixed Model

Sometimes it is not possible to gather all the information which is necessary to describe a population or, at least, to build a good model with the available covariates. These unobserved variables will make the model estimation process biased. Generalized linear mixed models purpose is to deal with such situations and to offer a solution to overcome the problem.

2.4.1 A brief overview

Generalized linear models assume the expected value of the response random variable is the transformation of the linear predictor. The underlying assumption is that all the necessary explicative variables have been observed. When this hypothesis is not met some bias is introduced in the model. Let us consider a simple linear model where the explicative variables are split into two groups, of which only one has been observed. This means the ith relation between the response and the regressors is

$$y_i = \alpha + \underline{x}_i^\intercal \beta + \underline{\omega}_i^\intercal b + \varepsilon_i,$$

where $\underline{x}_i^\intercal$ is the observed explicative variables vector, $\underline{\omega}_i^\intercal$ the unobserved regressors set and ε_i the error term for the ith subject. The maximum likelihood estimator for β, as shown in Section 2.1, is $\hat{\beta}_n = (X^\intercal X)^{-1} X^\intercal y$. With simple algebraic passages it can be shown that this solution is biased: the expected value of $\hat{\beta}_n$ is not β but $\beta + (X^\intercal X)^{-1} X^\intercal \Omega b$ where Ω is the matrix which columns are ω_i. The same problem affects generalized linear models. An overview of the problem and its solutions can be found in Pace and Salvan (1997, Section 9.3.3), and more detailed in Gelman and Hill (2007). This chapter will deal with just one solution that will be used in the next chapter.

2.4.2 The logistic mixed model

Section 2.3 has presented the logistic regression and it explained why it is a good model for binary responses so let us use it within the mixed model formulation. The response variable expected value is the inverse-logit transformation of the linear predictor, that is

$$\mathrm{E}[Y_i] = \pi_i = \frac{e^{\eta_i}}{1 + e^{\eta_i}},$$

where η_i is the linear predictor for the ith subject. The assumption is that the linear predictor is made up by two components, one well known ($\underline{x}_i^\intercal$) and one unknown ($b_i$)

$$\eta_i = \alpha + \underline{x}_i^\intercal \beta + b_i.$$

If all the b_i terms were known the likelihood function of the model would be

$$\mathcal{L}(\alpha, \beta) = \prod_{i=1}^{n} \left(\frac{e^{\eta_i}}{1 + e^{\eta_i}} \right)^{y_i} \left(\frac{1}{1 + e^{\eta_i}} \right)^{1-y_i},$$

but they are not because they are intrinsic of the subjects. The trick used by the generalized linear mixed models is treating the unknown terms as random variables with their own distribution function, $b_i \sim g(\cdot)$. The new likelihood function must take into account this new piece of information, that is

$$\mathcal{L}(\alpha, \beta) = \prod_{i=1}^{n} \left(\frac{e^{\eta_i}}{1 + e^{\eta_i}} \right)^{y_i} \left(\frac{1}{1 + e^{\eta_i}} \right)^{1-y_i} g(b_i).$$

On the other hand, the components b_i are not observed and therefore they are integrated out from the likelihood by integrating each term of the product in b_i, i.e.

$$\mathcal{L}_i^*(\alpha, \beta) = \int_{\mathbb{R}} \left(\frac{e^{\eta_i}}{1 + e^{\eta_i}} \right)^{y_i} \left(\frac{1}{1 + e^{\eta_i}} \right)^{1-y_i} g(b_i) db_i,$$

thus arriving at $\mathcal{L}(\alpha, \beta) = \prod_{i=1}^{n} \mathcal{L}_i^*(\alpha, \beta)$.

A typical assumption for $g(\cdot)$ is the normal distribution which implies the above integral cannot be solved analytically and some numerical approximation must be used. The next paragraph will present one possibility.

2.4.3 Laplace Approximation

The previous paragraph has introduced the logistic mixed model problem. It has been shown the likelihood function is expressed as the product of definite integrals. The computation of the analytical form of the primitive function is very often gruelling so some kind of approximation can be useful to save time. When the integrand can be expressed as the exponential of a continuous function with a unique absolute minimum a viable numeric approximation method of the definite integral is the Laplace approximation. Let us use the following notation

$$\int_{\mathbb{R}} e^{-nf(x)} dx,$$

where n, as usual, is the sample size and let $\tilde{x}$ be the unique absolute minimum of $f(\cdot)$. The Laplace approximation is computed from the Taylor expansion of $f(\cdot)$ around $\tilde{x}$, that is

$$\int_{\mathbb{R}} e^{-nf(x)} dx \doteq \int_{\mathbb{R}} \exp\Big\{ n \cdot f(\tilde{x}) - \frac{n}{2} f^{\mathrm{II}}(\tilde{x})(x - \tilde{x})^2 - \frac{n}{6} f^{\mathrm{III}}(\tilde{x})(x - \tilde{x})^3 + \\ - \frac{n}{24} f^{\mathrm{IV}}(\tilde{x})(x - \tilde{x})^4 + n \cdot O\left((x - \tilde{x})^5\right) \Big\} dx.$$

The term $nf^{\text{I}}(\tilde{x})(x-\tilde{x})$ does not appear in the previous formula due to $\tilde{x}$ definition. With some algebraic steps which involve the normal density function, see Pace and Salvan (1997, Section 9.3.3), the discussed integral can be approximated to

$$e^{-nf(\tilde{x})}\sqrt{\frac{2\pi}{nf^{\text{II}}(\tilde{x})}}\left(1+O(n^{-1})\right), \tag{2.3}$$

which is the general form of the Laplace approximation.

This thesis is focused on the logistic regression and some kind of generalization so, in the following lines the approximation will be applied to the logistic mixed model likelihood.

The first required step to be able to apply the Laplace approximation is writing the integrand as the exponential of a function with a unique absolute minimum. This can be easily done by involving the log-likelihood function which, by definition, is the natural logarithm of the likelihood function

$$\begin{aligned}\mathcal{L}_i^*(\alpha,\beta) &= \int_{\mathbb{R}} e^{\ell_i(\alpha,\beta)} \\ &= \int_{\mathbb{R}} \exp\log\left\{\left(\frac{e^{\eta_i}}{1+e^{\eta_i}}\right)^{y_i}\left(\frac{1}{1+e^{\eta_i}}\right)^{1-y_i} g(b_i)\right\} db_i \\ &= \int_{\mathbb{R}} \exp\left\{y_i\eta_i - \log\left(1-e^{\eta_i}\right) + \log g(b_i)\right\} db_i.\end{aligned} \tag{2.4}$$

The next step of the algorithm requires to find the absolute minimum (in b_i) of the exponent function. Before carrying out this step the distribution of the random effects must be known. Here we proceed with the generic function $g(\cdot)$ so the results hold in general. Usually the optimization has to be carried out numerically, using software for constrained maximum likelihood estimation.
The identified value, $\tilde{b}_i$, is then used to compute the Laplace approximation of the marginal likelihood which, for the logistic regression is

$$\ell_i^*(\alpha,\beta) \doteq \sum_{i=1}^{n} \tilde{\eta}_i y_i - \log\left(1+e^{\tilde{\eta}_i}\right) + \log g(\tilde{b}_i) - \frac{1}{2}\log|\tilde{j}_{b_i b_i}|,$$

where $\tilde{j}_{b_i b_i}$ is the ith diagonal element of the b-block of the likelihood observed information matrix when $b_i = \tilde{b}_i$. The parameter of interest estimates, i.e. $\hat{\alpha}$ and $\hat{\beta}$, are then computed from the approximated marginal likelihood using the standard estimation procedure of the likelihood theory, that is we compute its first derivative, we solve the likelihood equation and discriminate the roots with the help of the Hessian.

A useful software which handles random effects via the Laplace approximation is the `R` package `TMB`. This package creates a `C++` implementation of

the marginal likelihood and its analytical gradient from a `C++` template of the exponent function of (2.4) thus the optimization of the marginal likelihood from `R` is very easy and computationally non onerous. The use of this package is review in Chapter 5.

3 | Penalized Likelihood

Regression models are generally evaluated via their prediction error, defined as the mean square of the residuals

$$\mathrm{E}\left[(Y-\hat{Y})^2\right].$$

The prediction error can be split into two additive non-negative terms, the estimator variance and its squared bias

$$\mathrm{E}\left[(Y-\hat{Y})^2\right] = \mathrm{Var}\left[\hat{Y}\right] + \mathrm{B}\left(\hat{Y}\right)^2.$$

The bias of a an estimator is the difference between the estimated parameter true value and the mean of the estimator. In the case into account, the estimator is the prediction of Y, that is $\hat{Y}$, and the true value is Y itself. This means the squared bias factor of the current account is $\mathrm{B}^2(\hat{Y}) = \{Y - \mathrm{E}[\hat{Y}]\}^2$.

In regression models variance and bias trends are usually competitive, that is as the first grows the latter decreases. More formally, let p be the model complexity, usually given by the number of regressors, the estimator variance and bias have opposite trends: variance grows with the model complexity, on the other hand, bias diminishes with growing model complexity. An example of variance-bias trade-off is shown in Figure 3.1.

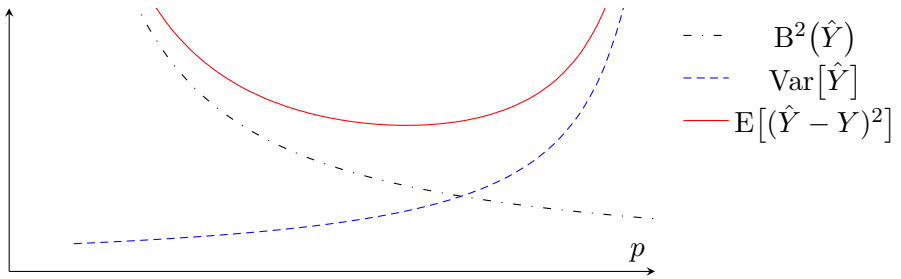


Figure 3.1: Variance-bias trade-off plot.

Usually model complexity is far smaller than sample size and this makes the prediction bias ignorable. The best model is the one which minimizes the predictor variance. On the other hand, when the model complexity is large,

variance increases without any important gain in bias. The model overfits the data involving an excess of optimism in evaluating the prediction error (Azzalini and Scarpa 2012).

Another problem that afflicts models when p is large is estimate instability. Estimate instability means little sample alterations cause great estimates changes. Finally, sometimes p is even greater than n, which makes the model non identifiable.

These problems can be overridden using some kind of regularization, that is modifying the estimation process in a way such that the new estimates are, in some sense, smoother than the default ones.

In generalized linear models a popular way is log-likelihood penalization by adding a term such that the solution of the penalized likelihood equations smooths the ordinary estimates, that is

$$\tilde{\ell}(\beta) = \ell(\beta) - s(\beta). \tag{3.1}$$

The term $s(\beta)$ has to decrease as the elements of β are smoother, usually uniformly closer to 0. This method is called *penalized likelihood* because it modifies the likelihood function by adding a penalization term which shrinks the maximum likelihood toward the target. Such technique has many benefits among which a prediction error reduction and estimate existence when it is affected by multicollinearity. The following paragraphs present the most popular penalization techniques. Further details can be found in Hastie et al. (2009, Sections 3.4, 3.8 and 7.1-7.3), Azzalini and Scarpa (2012, Chapter 3) and Agresti (2015, Chapter 11).

3.1 Ridge Penalization

Among the family of regularization methods, ridge regression introduces a quadratic penalization, that is $s(\beta) = \lambda\|\beta\|_2^2$ where $\|\cdot\|_2$ is the L_2 norm. Ridge estimates can be seen as the solutions of the maximization problem

$$\max_{\beta} \ell(\beta) \quad \text{subject to} \sum_{i=1}^{p} \beta_i^2 \leq t.$$

The Lagrangian form of the problem is actually

$$\max_{\beta} \left\{ \ell(\beta) - \lambda \sum_{i=1}^{p} \beta_i^2 \right\} = \max_{\beta} \tilde{\ell}(\beta) \tag{3.2}$$

which respects Equation (3.1) definition of penalized log-likelihood. The threshold t and the Lagrangian λ factor are bind by a one-to-one relation so the greater λ the greater estimate shrinkage. In linear models it is good practice letting the intercept unconstrained: its penalization would make the

procedure depending on the origin chosen for the Y response; that is, adding a constant c to each of the targets Y_i would affect the intercept as well as all the other coefficients. When dealing with a linear model, the problem described by the Equation (3.2) has an analytical solution,

$$\hat{\beta}_R = (X^\intercal X + \lambda I_p)^{-1} X^\intercal y,$$

where X is the regressors matrix and I_p the $p \times p$ identity matrix. It can be shown that leaving the intercept unconstrained leads to split the problem into two branches: estimating the intercept with the sample mean of the y data and estimating the other coefficients with a ridge regression of y against centred inputs. Under these hypothesis the estimates become:

$$\begin{cases} \hat{\beta}_0 = \bar{y} \\ \hat{\beta}_R = (\widetilde{X}^\intercal \widetilde{X} + \lambda I_p)^{-1} \widetilde{X}^\intercal y, \end{cases}$$

where $\widetilde{X}$ is the matrix made up by the last p columns of X, each one centred on its own sample mean, i.e. $\tilde{x}_i = x_i - \bar{x}_i$.

The choice of quadratic penalty $\|\beta\|_2^2$ makes ridge regression solution a linear function of y: it adds a positive constant to the diagonal of $X^\intercal X$ before inversion. The resulting problem is then non-singular, even if $X^\intercal X$ is not a full rank matrix.

3.2 LASSO

Another popular penalization solution is LASSO (Least Absolute Shrinkage and Selection Operator). Instead of penalizing the log-likelihood function with the square of euclidean distance of the β vector it uses the Manhattan norm of the β vector, that is $s(\beta) = \lambda \sum_{i=1}^{p} |\beta_i| = \lambda \|\beta\|_1$.

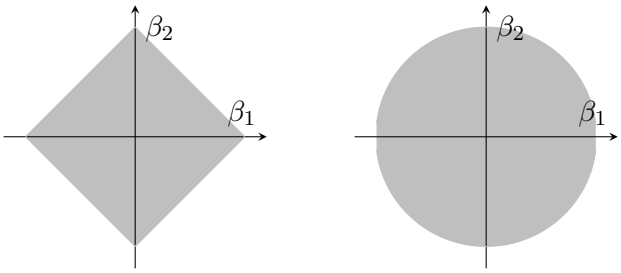
As ridge regression, LASSO regression can be defined as a constrained maximization problem:

$$\max_{\beta} \ell(\beta) \quad \text{subject to} \sum_{i=1}^{p} |\beta_i| \leq t.$$

Again, λ and t are bind by a one-to-one relation. As ridge Lagrangian form, LASSO Lagrangian form corresponds to a penalized likelihood:

$$\max_{\beta} \left\{ \ell(\beta) - \lambda \sum_{i=1}^{p} |\beta_i| \right\} = \max_{\beta} \tilde{\ell}(\beta). \tag{3.3}$$

Ridge and LASSO difference lies on the constrain, while ridge constrain shape is an hypersphere, LASSO's one is an hypercube. A graphical example when $p = 2$ is shown in Figure 3.2.



(a) LASSO constrain. (b) Ridge constrain.

Figure 3.2: LASSO and ridge constrains when $p = 2$.

LASSO constrain has sharp edges so some coefficients can be exactly zero: whenever the likelihood is tangent to a constrain edge, the coefficients laying on that edge are estimated as zero. This property is very useful because it provides an automatic variable selection method.

The λ parameter of (3.3) is called tuning parameter because it is used to set the shrinkage level: the greater λ the greater the shrinkage.

As with ridge regression, in linear models it is good practice leaving the intercept unconstrained. Again, the intercept estimate is the sample mean of the response while the other coefficients are estimated with a LASSO regression with the same response, that is y, against the centred regressors, that are $\tilde{x}_i = x_i - \bar{x}_i$. Where ridge regression estimates can be expressed analytically, LASSO estimates cannot, but efficient algorithms to estimate the model with a set of tuning parameter values have already been developed. For instance, Azzalini and Scarpa (2012) describe the LARS algorithm in Section 3.7, Hastie et al. (2015) present LARS and other algorithms.

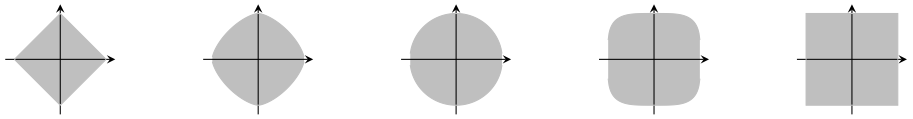
3.3 L_q penalization

Both ridge and LASSO penalization are proportional to a norm of the coefficients vector: the first one is the square of the euclidean norm, the second one is the Manhattan norm. However euclidean and Manhattan norm are just two possible choices. In linear algebra, a norm is a function that assigns a strictly positive length or size to each vector in a vector space, apart possibly for the zero vector, which is assigned a length of zero. This means that, for each real number $q \geq 1$, it is possible to define the q-norm as

$$\|x\|_q = \left(\sum_{i=1}^{p} |x_i|^q \right)^{\frac{1}{q}}.$$

From this specification it is clear that ridge and LASSO regression are just two possible choices. Using the generic q-norm as penalization function

is called L_q penalization, so LASSO il also called L_1 penalization and ridge is also known as L_2 penalization. Figure 3.3 shows the L_q constrain for different values of q.



(a) $q = 1$. (b) $1<q<2$. (c) $q = 2$. (d) $q > 2$. (e) $q \to \infty$.

Figure 3.3: L_q constrains when $p = 2$.

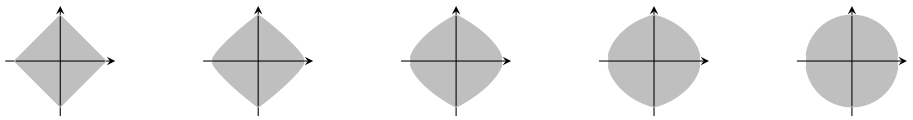
L_q penalization has been introduced to have more penalizing flexibility but it is more computationally onerous selecting the best pair (q, λ).

3.4 Elastic net

Another popular generalization of LASSO and ridge penalization is the elastic net penalization. It is defined as a weighted mean of the two penalizations:

$$s(\beta) = \lambda \sum_{i=1}^{p} \left(\alpha|\beta_i| + (1-\alpha)\beta_i^2 \right).$$

α is a number in $[0, 1]$. When α is 1, elastic net and LASSO penalization are the same, on the other hand, when α is 0, we get the ridge constrain. Mean value will define a constrain which lies between LASSO and ridge ones. Figure 3.4 shows elastic net constrain with different α values when $p = 2$.



(a) $\alpha = 1$. (b) $\alpha = 0.75$. (c) $\alpha = 0.5$. (d) $\alpha = 0.25$. (e) $\alpha = 0$.

Figure 3.4: Elastic net constrains.

While α role is defining the constraint shape, λ role is setting the coefficient shrinkage level: the greater λ the greater the shrinkage.

Elastic net constraints are very similar to L_q constraints but, while the second one has differentiable edges, the first has not. This means elastic net can set some coefficients exactly to zero while L_q can not. Elastic net provides a more flexible approach than ridge and LASSO regression but it has two tuning parameters, λ and α, instead of one. This makes elastic net tuning more onerous.

4 | Item-Response Theory

IRT is the acronym for *Item Response Theory*. The purpose of IRT is to model the probability that a subject would answer correctly a specified amount of questions, also called items, given the total number of tests. This model was proposed the first time by Rasch, so some IRT models are also known as Rasch models (Fienberg 2004). The first paragraph describes the typical setting while the following introduce the most used models, the 1PL and 2PL.

4.1 Model setup

The purpose of IRT is to model the probability that a subject right-answers an item with a function which depends on some parameters. Each subject has to answer each item just once so a suitable model which describes the problem is through Bernoulli random variables. The probability parameter of these variables depends on the subject and the item. The whole problem can be analytically expressed as

$$\begin{aligned} Y_{si} &\sim \text{Be}(\pi_{si}), \\ \pi_{si} &= f(\theta, s, i), \end{aligned} \tag{4.1}$$

with $s = 1, \dots S$ and $i = 1, \dots I$. The s subscript is the index of the subject while i is the item index. From this follows that I is the total amount of items, S the number of subjects and $n = S \cdot I$ is the total sample size. For now $f(\cdot)$ is a generic function which summarizes the subject-item couple with a number from 0 to 1. Finally Y_{si} is 1 when subject s gives the right answer to item i, and 0 otherwise.

The model in (4.1) is a generalized linear model for binary data thus the

log-likelihood function associated to the given problem is the following

$$\begin{aligned}\ell(\theta) &= \sum_{s=1}^{S}\sum_{i=1}^{I} y_{si}\log\pi_{si} + (1-y_{si})\log(1-\pi_{si}) \\ &= \sum_{s=1}^{S}\sum_{i=1}^{I} y_{si}\log f(\theta,s,i) + (1-y_{si})\log\big(1-f(\theta,s,i)\big) \\ &= \sum_{s=1}^{S}\sum_{i=1}^{I} y_{si}\operatorname{logit} f(\theta,s,i) + \log\big(1-f(\theta,s,i)\big).\end{aligned} \tag{4.2}$$

The θ parameter represents the vector of all parameters used by the model. The following sections will provide a more detailed description of the model, depending on the specific assumptions.

4.1.1 1PL model

The simplest parametric model is a logistic regression where the response variable is Y_{si} and the regressors are subject and item indicators. The link function $f(\cdot)$ is the inverse of the logistic function, which is the canonical choice for binomial data. Formally, the problem can be expressed as

$$\begin{aligned} Y_{si} &\sim \text{Be}(\pi_{si}), \\ \pi_{si} &= \frac{e^{\gamma_s+\alpha_i}}{1+e^{\gamma_s+\alpha_i}}. \end{aligned} \tag{4.3}$$

From Equation (4.2) we have

$$\ell(\alpha,\gamma) = \sum_{s=1}^{S}\sum_{i=1}^{I} y_{si}(\gamma_s+\alpha_i) - \log\big(1+e^{\gamma_s+\alpha_i}\big). \tag{4.4}$$

This is a classical logistic regression and can be easily estimated with the common generalized linear model estimating algorithm. Each α_i parameter represents the difficulty of that item while each γ_s represents the ability of that subject. The parameter $\alpha = (\alpha_1, \dots \alpha_I)$ can be estimated by means of conditional likelihood, thus eliminating the effect of nuisance parameter $\gamma = (\gamma_1, \cdots \gamma_S)$. Of course, we can switch the role of α and γ for the estimation of γ. We note that the conditional likelihhod is valid also when one assumes the γ_s as random effects (Sartori and Severini 2004).

The model of (4.4) is non identifiable without any constrain. It is common practice setting one of the α_i coefficients, i.e. α_1 equal to 0 or their sum equal to 0, i.e. $\sum_{i=1}^{I}\alpha_i = 0$ and similarly for the γ_s coefficients, that is $\gamma_1 = 0$ or $\sum_{s=1}^{S}\gamma_s = 0$.

4.1.2 2PL model

The difference between the 2PL and the 1PL model is the addition of a new vector of coefficients which purpose is quantifying the item discrimination power, that is the how much that item discriminates the subjects by their ability. These new coefficients will be denoted by $\beta = (\beta_1, \dots \beta_I)$. The new probability of success is then

$$\begin{aligned} Y_{si} &\sim \text{Be}(\pi_{si}), \\ \pi_{si} &= \frac{e^{\beta_i \gamma_s + \alpha_i}}{1 + e^{\beta_i \gamma_s + \alpha_i}}. \end{aligned} \tag{4.5}$$

Note that $\beta_i = 1$, $i = 1, \dots I$ gives the 1PL model. This new model likelihood can be easily written by adding the β_i coefficients in Equation (4.4) as shown in (4.6).

$$\ell(\alpha, \beta, \gamma, y) = \sum_{s=1}^{S} \sum_{i=1}^{I} y_{si}(\beta_i \gamma_s + \alpha_i) - \log\bigl(1 + e^{\beta_i \gamma_s + \alpha_i}\bigr). \tag{4.6}$$

The new linear predictor is non-linear in $\theta = (\alpha, \beta, \gamma)$, the vector of coefficients, as opposed to the linear predictor of the 1PL model. This means this model cannot be fitted with logistic regression software.

Equation (4.5) is not the only parametrization of the model. Another common parametrization is

$$\pi_{si} = \frac{e^{\beta_i(\gamma_s + \tilde{\alpha}_i)}}{1 + e^{\beta_i(\gamma_s + \tilde{\alpha}_i)}},$$

where $\tilde{\alpha}_i = \frac{\alpha_i}{\beta_i}$. Even this parametrization is non-linear in θ. Among these two different parametrizations this thesis will use the one in (4.5).

4.2 Penalized likelihood for IRT models

The 1PL model, when I and S are fixed, can be represented as a generalized linear model so its estimation is possible, the 2PL model is non linear in the θ vector so its estimation is gruelling. Even numeric maximization algorithms do not converge because, although the model is identifiable, the parameter estimates are unstable and they typically do not converge even using great amount of data. To overcome the described situation, this thesis proposes to proceed as follows:

1. 2PL models, in literature, are almost always estimated using random effects for the subject parameters so we treat the γ_s coefficients like random terms of a logistic mixed model and marginalize with respect to the γ_s coefficients with the Laplace approximation;

2. Even though assuming the γ_s as random effects allows to obtain maximum likelihood estimate of the item parameters, these are not always satisfactory. Therefore we also use some kind of penalized likelihood, penalizing the β_i coefficients, on the approximated likelihood in order to obtain a compromise between 1PL and 2PL models.

Each step is analysed in the following paragraphs.

4.2.1 Laplace approximation of the 2PL likelihood function

Section 2.4 discussed generalized linear mixed models and showed how the likelihood function has to be modified to take into account the information brought by the random terms. Such information can be included only if the random effects distribution is known. A common approach, which this thesis follows, is assuming the random coefficients are independent and identically distributed as a Gaussian random variable. The assumption the $\gamma_s \sim N(0,1)$ avoids us the constrains on α and β that are needed when the γ_s are treated as fixed effects. With this piece of information, each term of the likelihood function becomes

$$\mathcal{L}_s(\alpha,\beta;y_s) = \int_{\mathbb{R}} \prod_{i=1}^{I} \left\{ \left(\frac{e^{\eta_{si}}}{1+e^{\eta_{si}}} \right)^{y_{si}} \left(\frac{1}{1+e^{\eta_{si}}} \right)^{1-y_{si}} \right\} \phi(\gamma_s) d\gamma_s \tag{4.7}$$

where $\eta_{si} = \alpha_i + \beta_i \gamma_s$ and $\phi(\cdot)$ is the standard normal density function. As (4.7) shows, the terms of the likelihood function are definite integrals in γ_s and they can be written in the form $\int_{\mathbb{R}} e^{f(x)} dx$ so their value can be approximated with the Laplace formula. The optimum of the exponent part of the integrand function of

$$\int_{\mathbb{R}} \exp \left\{ \sum_{i=1}^{I} \eta_{si} y_{si} - \sum_{i=1}^{I} \log\left(1+e^{\eta_{si}}\right) + \log \phi(\gamma_s) \right\} d\gamma_s,$$

i.e. $\tilde{\gamma}_s$, is used to obtain the approximation

$$\ell_s^*(\alpha,\beta,y_s) \doteq \sum_{i=1}^{I} \tilde{\eta}_s y_s - \sum_{i=1}^{I} \log\left(1+e^{\tilde{\eta}_s}\right) + \log \phi(\tilde{\gamma}_s) - \frac{1}{2} \log \tilde{j}_{\gamma_s \gamma_s}$$

where $\tilde{j}_{\gamma_s\gamma_s}$ is the sth element of the γ-block of the penalized likelihood observed information matrix when $\gamma_s = \tilde{\gamma}_s$.

This step will be performed using software `TMB`, as described in the next chapter.

4.2.2 Penalized 2PL model

The 1PL model can be easily estimated due to its link with the logistic regression while the 2PL model cannot because the linearity hypothesis is not met. From this consideration bears the idea of penalizing the β coefficients in the likelihood function with the purpose of making the 2PL model more regular and more similar to the 1PL one. The penalty used in this thesis is the L_1 in the hope that only a few non-one β coefficients will be estimated. The resulting marginal penalized log-likelihood function is

$$\tilde{\ell}(\alpha, \beta, y) = \sum_{s=1}^{S} \sum_{i=1}^{I} \Bigg(\tilde{\eta}_{si} y_{si} - \log\left(1 + e^{\tilde{\eta}_{si}}\right) + \log \phi(\tilde{\gamma}_s) + \\ -\frac{1}{2} \log \tilde{j}_{\gamma_s \gamma_s} \Bigg) - \lambda \sum_{i=1}^{I} |\beta_i - 1|, \tag{4.8}$$

where the λ parameter must be specified.

In Section 3.2 λ was called tuning parameter because it sets the shrinkage level of the model. Its selection is important because if the chosen value is too big it will introduce unneeded bias while too small values will not reduce estimates variation enough. There are many selection methods; this thesis explores the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC). Both methods belong to the family of model selection based on information criteria. Such methods select the tuning parameter which minimizes the expression

$$\mathrm{IC}(\lambda) = -2\,\ell(\hat{\theta}(\lambda), \lambda) + \mathrm{penalty}(p),$$

with $\hat{\theta}(\lambda)$ the estimate of (α, β) obtained from (4.8) with the given λ, and the penalty function is a positive factor which is directly proportional to the number of non-zero estimates: the greater that number, the greater the penalty. Akaike's penalty is the double of the number of non-zero coefficients, that is $\mathrm{penalty}(p) = 2p$. The Bayesian Information Criterion penalty is the number of non-zero estimates multiplied by the natural logarithm of the sample size, that is $\mathrm{penalty}(p) = p \log n$. The two criteria have a similar structure but their theoretical base is very different: while the first one has been extracted from the theoretical expected distance between the true-model and the estimated model likelihood, the latter is got from a Bayesian perspective. A more detailed description of model selection based on information criteria can be found in Azzalini and Scarpa (2012, Section 3.5.3) and Hastie et al. (2009, Section 7.5).

5 | Simulation Study

The purpose of this thesis is analysing the properties of the α and β coefficient estimators. The estimators considered are:

1. marginal penalized maximum likelihood estimators;
2. marginal penalized maximum likelihood estimators computed via L_1-penalized marginal likelihood. This scenario is further split into two scenarios depending on the method used to select the tuning parameter:
   (a) tuning parameter selection via AIC;
   (b) tuning parameter selection via BIC.

The estimators are computed in simulated samples. We chose to use samples made up by a quite large number of subjects (500) and a relatively large number of items (30). It's obvious the term large must be read relatively to context.

All the models assume random effects for the subject parameters, i.e. the γ coefficients: in literature it is common practice when dealing with 2PL models and we are interested on the item properties, the α and β coefficients, rather than the subject abilities, the γ vector. The difference between the likelihood function of Steps 1. and 2. above lies on the L_1 penalty introduced in the second one: the model underlying Step 1. is the one presented in Section 4.2.1 while Steps 2.(a) and 2.(b) use the one proposed in Section 4.2.2. The choice of using an L_1 penalty bears from two considerations:

- the 2PL estimation is gruelling even using numerical algorithms;
- only a limited subset of the β coefficients is really significant which implies the 2PL model is too complex while the 1PL does not fit the data well enough. The L_1 penalty should allow to focus on the non-one β parameters.

We carry out the selection of the L_1 tuning parameter using the Akaike and Bayesian information criteria. Among the objective of this thesis is studying which of the two criteria selects the best λ, that is the one which identifies the greatest number of β coefficients truly equal to one.

The α and β estimator properties which we want to study are their bias and their mean square error. We want to see which is the best tuning parameter selection method too.

5.1 Simulation Structure

The results presented in this thesis have been achieved via simulation using `R`, version 3.2.2 (R Core Team 2014). The simulation code is reported in Appendix. This section illustrates the simulation process, in particular Section 5.1.1 explains the sampling procedure and Section 5.1.2 the estimation algorithm.

5.1.1 Sample Generation

The simulation has been run through 500 samples, each of which has been sampled from the same set of coefficients. The sample size has been fixed to 500 subjects (i.e. $S = 500$) and 30 items (i.e. $I = 30$) and every subject-item couple (i.e. y_{si}) is the realization of a Bernoulli random variable indexed by a parameter which is the inverse-logistic of the linear predictor, that is $\pi_{si} = \frac{e^{\eta_{si}}}{1+e^{\eta_{si}}}$. The α vector is the realization of a uniform random variable, the γ array has been sampled from a standard normal variable while the β coefficients have been arbitrarily chosen: half of them were set to 1, another quarter were set to 0.5 and the last quarter were set to 1.5.

5.1.2 Estimation Procedure

For each sample we carried out the following steps:

1. Fix a grid of λ values;
2. Maximize the penalized log-likelihood presented in (4.8) for each λ value;
3. Save the λ values which minimize either the AIC or the BIC;

The grid of λ values must be the same for all the datasets. This thesis has used 200 equispaced values from 0 to 200. Step 2 has been carried out using two external packages, `TMB` and `lbfgs` which use is explained in the following sections.

`lbfgs` Package

Package `lbfgs` (Coppola et al. 2014) is a general optimisation package which does not belong to the basic version of `R` but it must be installed from the CRAN. We chose it rather than other built-in packages mainly for the following considerations:

- it manages L_1 penalizations through the specification of the tuning parameter without further coding;
- the L_1 penalization can be applied to any subset of the parameter vector;
- it makes use of the gradient even when L_1 penalization are used.

The last one and other considerations made us using the `TMB` package which purpose and usage are explained in the next section. Further information about `lbfgs` usage can be found in the user manual from the CRAN repository or from the help accessible from `R`.

`TMB` Package

`TMB` is a package which purpose is offering a simple interface between `R` and `C++` and managing random effect models via Laplace approximation (Kristensen et al. 2015). With this package the likelihood maximization steps are:

1. write a `C++` model template, i.e. a `.cpp` file;
2. compile the template from `R`;
3. load the compiled file from `R`;
4. create an object from the loaded environment with the observed data and the compiled function;
5. optimize the function contained in the created object with any optimizer in `R`.

The template described in Step 1 is a `C++` file containing the instruction to compute the function which has to be optimized. `R` built-in functions, such as density functions, can be used within the template. The compilation of the template from `R` is handled by the method `compile` of this package.
Step 3 is carried out by the built-in `dyn.load` method but it should be called on `dynlib(<file_name>)`: the method `dynlib` adds the platform dependent dynamic libraries extensions so the code can be used cross-platform.
Step 4 is carried out by the function `MakeADFun`. It needs the following arguments:

- a list containing all the workspace objects used by the template, i.e. the number of items, the number of subjects and the observed data;
- the parameters of the function, i.e. α, β and γ vectors;
- the name of the template file (without the extension);

- the name of the workspace object which contains the random effect that will be marginalized through the Laplace approximation, that is the γ vector.

The created object contains both the Laplace approximation of the marginal likelihood function and its analytical gradient computed using automatic differentiation. We carried out Step 5 with the package `lbfgs` which has already been presented. There are two great advantages of using `TMB`: first the Laplace approximation is automatically computed starting from the integrand function and it is implemented in `C++`; secondly the gradient is available analytically and therefore `lbfgs` can completely exploit its power.

Further information about `TMB` usage can be found in Kristensen et al. (2015).

5.2 Simulation Results

In the previous section we have illustrated the simulation setup we have followed, here we present and analyse the results we have collected from such simulation.

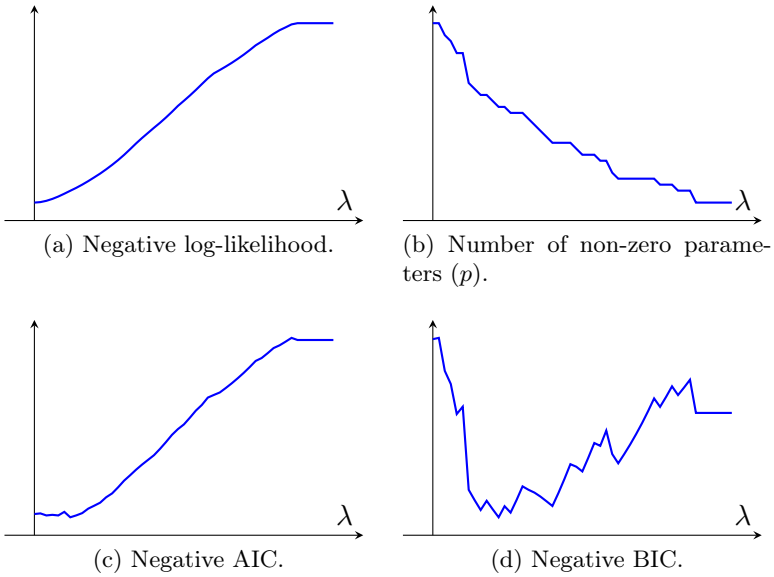
5.2.1 Single Sample Analysis

A random subset of the generated samples has been taken into account to check the log-likelihood, AIC and BIC trends. An example is reported in Figure 5.1. It can be seen the negative maximized log-likelihood and the number of non-zero estimates have opposite trends as function of λ: the former is increasing while the latter is decreasing. The AIC and BIC trends, which are linear combination of the previous two, are decreasing and then increasing. These observations match the theoretical expectations: the larger λ value the greater the shrinkage the fewer the number of non-zero-coefficients and the lower the likelihood of the chosen parameter set. Due to their trend the AIC and the BIC have an absolute minimum, the λ value corresponding to such value is the one used to estimate the model.

5.2.2 Aggregated Results

Section 5.1.2 has presented the procedure used to select the best λ value given a specific sample. The algorithm identifies two values: the one which minimizes the AIC curve and the one which minimizes the BIC curve. The estimates computed with such λ values are the model estimates. This paragraph analyses their properties via the computation of some statistics like the observed bias and standard deviation and with the assistance of some plots.

We started our analyses from the inspection of the cumulative bias and standard deviation of the estimates. The former is the cumulative mean bias



(a) Negative log-likelihood.

(b) Number of non-zero parameters (p).

(c) Negative AIC.

(d) Negative BIC.

Figure 5.1: Example of Likelihood, p, AIC and BIC profiles.

of the coefficients, that is $\bar{B}_\theta = \sum_{i=1}^{p} \bar{B}(\theta_i)$, the latter is the square root of the cumulative variance of the coefficients, that is $\bar{\sigma}_\theta = \sqrt{\sum_{i=1}^{p} \sigma_i^2}$. Table 5.1 reports this two statistics for each coefficient group: the first three columns contain the quantities of the α coefficients when they have been estimated via maximum likelihood, with the λ value selected via AIC and BIC while the latter three contain the same quantities referred to the β coefficients.

	α			β		
	MLE	AIC	BIC	MLE	AIC	BIC
$\bar{B}_\theta$	3.39	3.40	3.34	0.18	1.15	3.49
$\bar{\sigma}_\theta$	0.57	0.56	0.56	0.76	0.66	0.61

Table 5.1: Cumulative Bias and Standard Deviation of the estimators of α and β.

We observe bias and standard deviation of the estimators of the α coefficients are more or less the same through the three scenarios: the penalty applied to the likelihood regards the β coefficients only so the α statistics are more or less constant through the different scenarios. On the other hand the β coefficients bias and standard deviation are not flat and they are somehow inverse proportional: the maximum likelihood method reaches the smallest bias but has the greatest standard deviation, on the other hand, the BIC reaches the smallest standard deviation but the greatest bias.

Of course the aggregated measure is not informative of the single components of the estimator. This is done graphically using box plots. Figures 5.2 and 5.3 report respectively the box plot of the estimates of the α and β coefficients we obtained with maximum likelihood, AIC and BIC penalizations.

We see the estimators for α are biased and no method seems to be able to correct the problem. The estimators for β on the other hand are influenced by the chosen method. Figure 5.3a shows that the estimates from the maximum likelihood estimator are unbiased and their variance is more ore less constant. Figure 5.3c shows that the estimates from the BIC have different behaviors: the coefficients which true value is 1 are unbiased and their variance is very small while the significant ones are biased and their variance seems greater than the one of the estimates obtained from the maximum likelihood procedure. The AIC is a compromise: in Figure 5.3b all the boxes pass through zero and their size is placed between the size of the maximum likelihood and BIC ones.

From the previous consideration we have seen the main impact on the estimates involve their bias so we have done a further focus on it. Figure 5.4 reports the average bias of the α coefficients and Figure 5.5 reports the same statistics computed for the β coefficients.

The three methods are almost equivalent when we want to estimate the α coefficients but they are quite different when dealing with the β coefficients: the BIC reaches the smallest bias for the non significant β while the maximum likelihood estimates are the best one to estimate the significant ones. The AIC estimates are always a middle point between the maximum likelihood and the BIC ones but for the non significant β. This must be imputed to the different penalty used by the two criteria: the AIC uses a less strong penalty than BIC thus it selects smaller λ values so the distance between the AIC estimates and the maximum likelihood ones is smaller than the distance between the BIC estimates and the maximum likelihood ones.

Among our objectives we wanted to understand which of AIC and BIC is the best method to identify the non-significant β coefficients. Table 5.2 reports the average bias of the number of non-zero estimated coefficients for the three methods. From this table and Figures 5.5b and 5.3 we can say

	MLE	AIC	BIC
Bias	15	8.92	−0.65

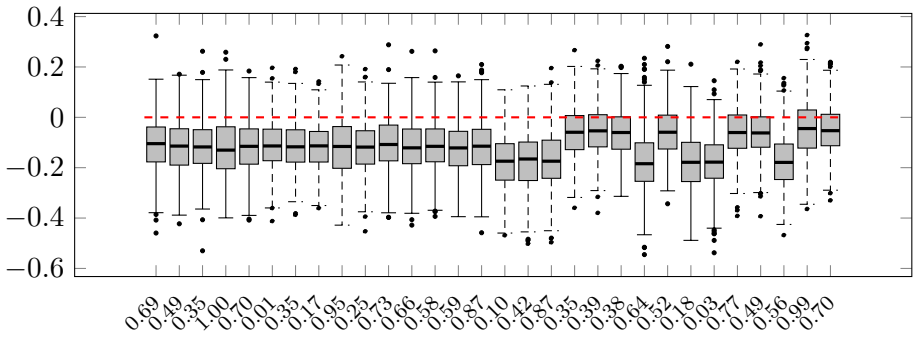
Table 5.2: Bias of the number of non-zero coefficients (p).

the BIC is the best method to be used to identify which coefficients are significant and which are not. From Figure 5.3 we see the BIC boxes of the bias of the non significant coefficients are collapsed on 0 and Figure 5.5b shows the average bias of such coefficient is minimized by the BIC line. In the

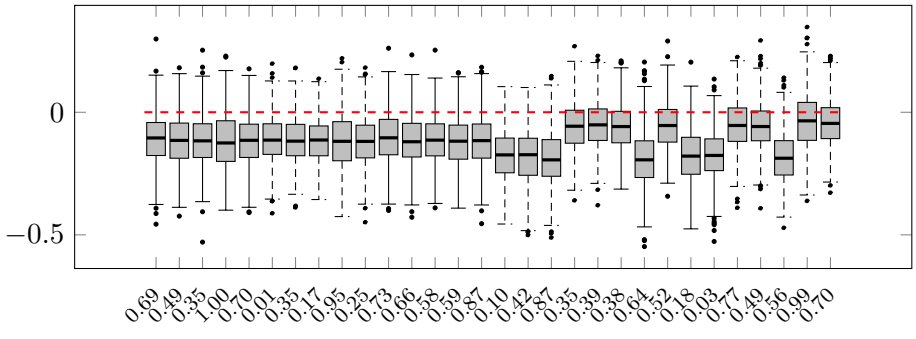
end Table 5.2, which reports the bias of the number of non-zero coefficients reaches its minimum in correspondence to the BIC column.

Summing up the analysis we conclude by saying the penalization of the β coefficients reduces the bias and variance of the non-significant β coefficients but this makes the same indexes a bit worse for the significant ones. The best value of the tuning parameter to identify the non significant coefficient is selected by the BIC while the AIC one is less good at screening the coefficients but introduces less bias.

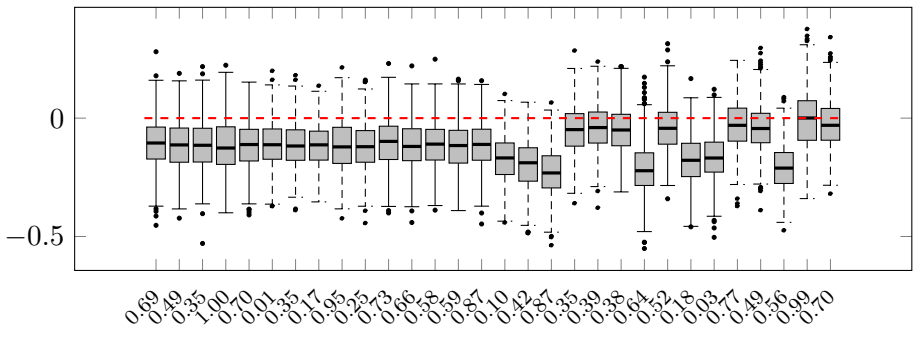
Of course these are only preliminary results and more research is needed in order to investigate the behavior of different estimators in different configurations, with different values of I, S and the true number of β values not equal to 1.



(a) Maximum Likelihood Estimates.

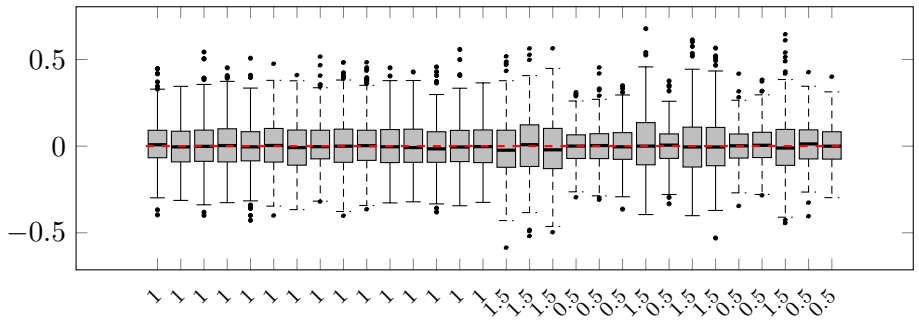


(b) AIC estimates.

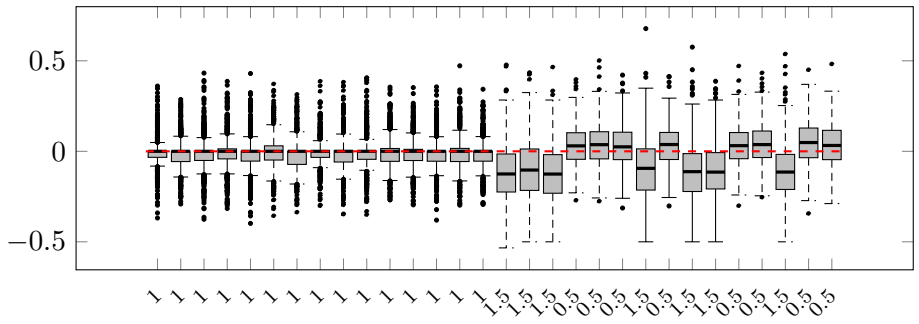


(c) BIC estimates.

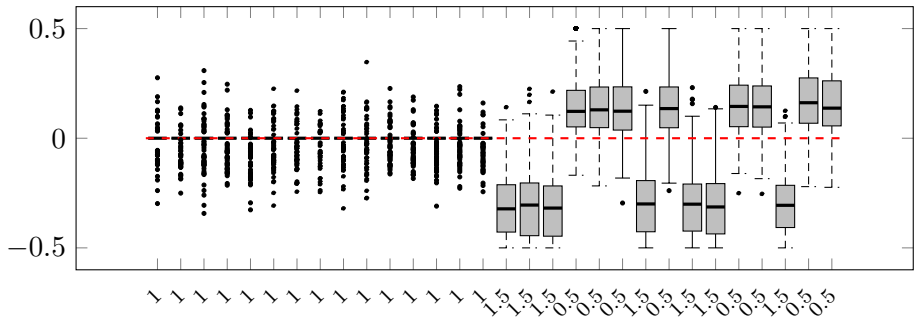
Figure 5.2: α estimates box-plots.



(a) Maximum Likelihood Estimates.

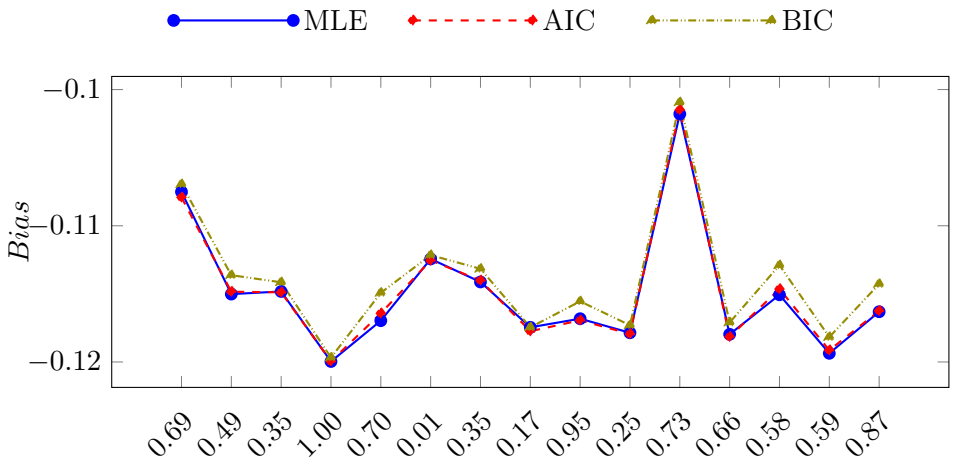


(b) AIC estimates.

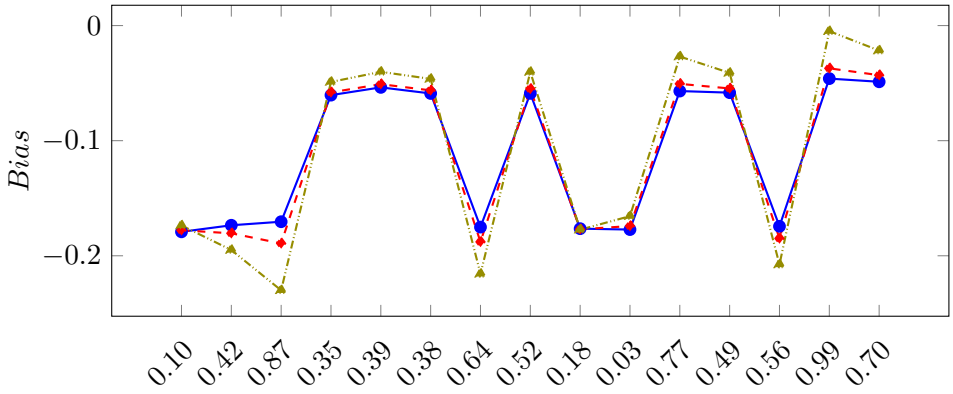


(c) BIC estimates.

Figure 5.3: β estimates box-plots.

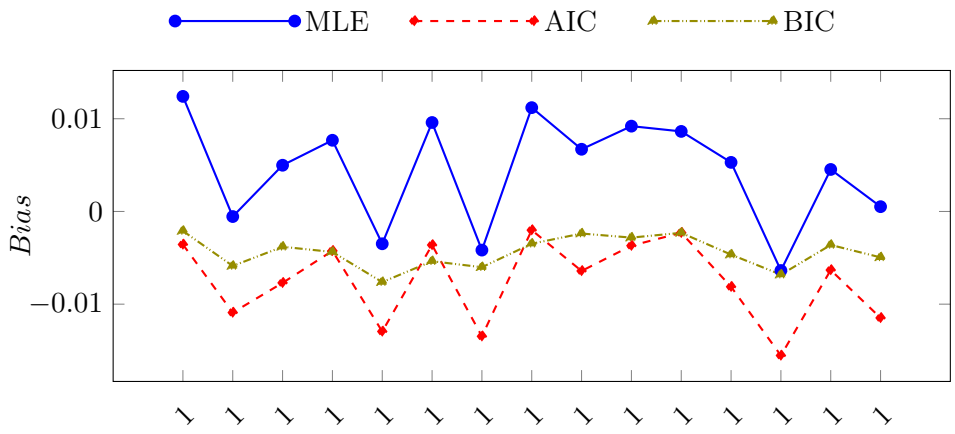


(b) α_1 - α_{15} coefficients bias.

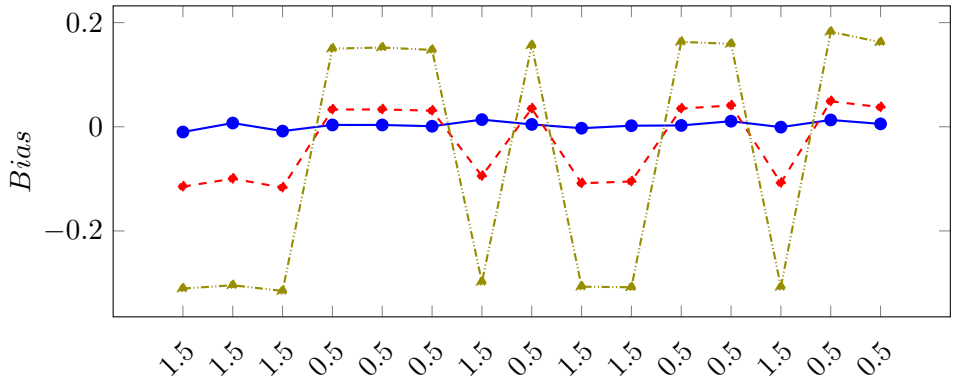


(c) α_{16} - α_{30} coefficients bias.

Figure 5.4: α coefficients bias.



(b) β_1 - β_{15} coefficients bias.



(c) β_{16} - β_{30} coefficients bias.

Figure 5.5: β coefficients bias.

Conclusions

In this thesis we have review the Item-Response Theory and two of its model, the 1PL and the 2PL. The first one can be estimated due to its link with the generalized linear models and the logistic regression in particular while the 2PL model estimation is more gruelling because even using numeric algorithms and huge amount of data the parameter estimates do not converge. Our purpose is exploring a new way to estimate the 2PL model.

After reviewing likelihood methods and IRT models in Chapters 1 and 4, we focused on penalization methods for 2PL models with subject specific random effects. The aim was to compare standard marginal likelihood with penalized versions of it, making a compromise between 1PL and 2PL models.

To this aim we have considered a simulation through a set of 500 samples each of which has been extracted from the same model. For each sample we have estimated the 2PL model with random effects with and without penalizations on the likelihood function. We carried out the selection of the tuning parameter using the Akaike and Bayesian information criteria. Among our purposes we wanted to detect which of the two methods should be used to get the best selection of significant coefficients.

The analysis of the collected results makes us reckon the use of a penalty on the marginal likelihood is useful to select which coefficients are really significant and in such sense the BIC selects the best value of the tuning parameter, that is the penalization level. On the other hand we have not detected any particular gain in the estimates bias and just a little shrinkage of their standard deviation.

Of course these are only preliminary results and more research is needed in order to investigate the behavior of different estimators in different configurations with different values of I, S and the true number of β values not equal to 1. Further investigation is needed, for instance, because we have observed a strange bias for the estimators for α independently on the estimation procedure.

The shrinkage applied on the β coefficients introduces bias on the other parameters thus identifying which β coefficient are really in the model and then estimating the restricted model with maximum likelihood is an interesting procedure which could be considered.

We have used AIC and BIC criteria in order to select the best value of the tuning parameter but other procedures, such as the cross validation, should be inspected. It could be interesting to change the penalty function as well trying, for instance, the ridge and the elastic net penalties.

Finally it is worthy inspecting the inferential properties of the model as well. For instance, a simulation study of the empirical coverage level of the Wald confidence intervals for components of the parameter should be considered.

Bibliography

Agresti, A. (2015). *Foundations of Linear and Generalized Linear Models*. English. Ed. by Wiley. Wiley Series in Probability and Statistics. Hoboken, New Jersey: Wiley.

Azzalini, A. (1996). *Statistical Inference Based on the likelihood*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis.

Azzalini, A. and Scarpa, B. (2012). *Data Analysis and Data Mining: An Introduction*. Oxford University Press, USA.

Baker, F. B. (1987). "Methodology Review: Item Parameter Estimation Under the One-, Two-, and Three-Parameter Logistic Models". In: *Applied Psychological Measurement* 11.2, pp. 111–141.

Coppola, A., Stewart, B., and Okazaki, N. (2014). *lbfgs: Limited-memory BFGS Optimization*. R package version 1.2.1.

Fienberg, S. E. (2004). "Rasch model". In: Kotz, S. and Johnson, N.L. *Encyclopedia of statistical sciences*. Ed. by C.B. Read. New York: Wiley.

Fisher, R. A. (1922). "On the Mathematical Foundations of Theoretical Statistics". In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 222.594-604, pp. 309–368.

Gelman, A. and Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Analytical Methods for Social Research. Cambridge University Press.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer series in statistics. New York: Springer.

Hastie, T. J., Tibshirani, R., and Wainwright, M. (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis.

Kristensen, K., Nielsen, A., Berg, C. W., Skaug, H. J., and Bell, B. (2015). *TMB: Automatic Differentiation and Laplace Approximation*. ArXiv e-print; in press, *Journal of Statistical Software*.

Pace, L. and Salvan, A. (1997). *Principles of Statistical inference from a Neo-Fisherian Perspective*. Singapore: World Scientific Publishing.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria.

Sartori, N. and Severini, T. A. (2004). "Conditional likelihood inference in generalized linear mixed models". In: *Statistica Sinica*, pp. 349–360.

A | Source Code

Listing A.1: Used functions.

```
# +----------------------------------------+ #
# | > Likelihood functions                 | #
# +----------------------------------------+ #
p.range = function(p, eps=2.22e-15) {
  out = p;
  out[p<eps] = eps;
  out[p>(1-eps)] = (1-eps);
  return(out);
}



tilde_ell_s = function(gamma,alpha,beta,y) {
  # penalized log likelihood for the s-th subject/
     category
  etas = alpha+beta*gamma
  ps = p.range(plogis(etas))
  sum(y*log(ps)+(1-y)*log(1-ps))-0.5*gamma^2
}



marg2PL_Laplace = function(param,data) {
  # param is (alpha,beta-1), of dimension 2*I
  # data is Kx(I+1) where:
  #    - the rows are the distinct configurations
  #    - the first I columns represents y for the
     items
  #    - last column has the frequencies of the
     configurations
  I = ncol(data)-1
  K = nrow(data)
  freq = data[,I+1]
  y = data[,1:I]
  alpha = param[1:I]
  beta = param[-(1:I)]+1
```

```
34    out = 0
35    for (k in 1:K)
36    {
37      app = nlminb(0,function(x) -tilde_ell_s(x,alpha,
          beta,y[k,]))
38      #   cat(app$par,app$objective,"\n",sep=" ")
39      pk = p.range(plogis(alpha+beta*app$par))
40      jkk = sum(beta^2*pk*(1-pk))+1
41      # print(jkk)
42      # print(optimHess(app$par,tilde_ell_s,alpha=alpha
          ,beta=beta,y=y[k,]))
43      out = out+freq[k]*(-app$objective-0.5*log(jkk))
44    }
45    out
46  }
47
48
49
50  marg2PL_Laplace_grad = function(param,data) {
51    # param is (alpha,beta-1), of dimension 2*I
52    # data is Kx(I+1) where:
53    #     - the rows are the distinct configurations
54    #     - the first I columns represents y for the
        items
55    #     - last column has the frequencies of the
        configurations
56    I = ncol(data)-1
57    K = nrow(data)
58    freq = data[,I+1]
59    y = data[,1:I]
60    alpha = param[1:I]
61    beta = param[-(1:I)]+1
62    out = rep(0,2*I)
63    for (k in 1:K) {
64      app = nlminb(0,function(x) -tilde_ell_s(x,alpha,
          beta,y[k,]))
65      gammatilde = app$par
66      pk = p.range(plogis(alpha+beta*gammatilde))
67      jkk = sum(beta^2*pk*(1-pk))+1
68      jkalpha = beta*pk*(1-pk)
69      jkbeta = beta*gammatilde*pk*(1-pk)-(y[k,]-pk)
70      ellalpha = y[k,]-pk
71      ellbeta = gammatilde*(y[k,]-pk)
72      jkkalpha = beta^2*pk*(1-pk)*(1-2*pk)
73      jkkbeta = beta^2*gammatilde*pk*(1-pk)*(1-2*pk)+2*
          beta*pk*(1-pk)
74      jkkk = sum(beta^3*pk*(1-pk)*(1-2*pk))
75      dgam.dalpha = -jkalpha/jkk
76      dgam.dbeta = -jkbeta/jkk
```

```
77      out[1:I] = out[1:I]+freq[k]*(ellalpha-(0.5/jkk)*(
            jkkalpha+dgam.dalpha*jkkk))
78      out[-(1:I)] = out[-(1:I)]+freq[k]*(ellbeta-(0.5/
            jkk)*(jkkbeta+dgam.dbeta*jkkk))
79    }
80    out
81  }
```

Listing A.2: Datasets generation.

```
1  # +----------------------------------------+ #
2  # | > Data generation                      | #
3  # +----------------------------------------+ #
4  # string manipulation library
5  library(stringr);
6
7  # number of random datasets
8  N = 500;
9
10 # set the random generator seed
11 set.seed(777);
12
13 # set parameters
14 I =  30;
15 S =  500;
16
17 # file names
18 data.dir = "data/";
19 names.full = rep(NA, N);
20 names.comp = rep(NA, N);
21
22 # data generation log file
23 LOG_FILE = paste(data.dir,"data-generation.log", sep=
       "");
24 # clear the file
25 cat(as.character.Date(Sys.time()), "\n\n", file=
       LOG_FILE);
26 # should the coefficients be printed?
27 print.param = TRUE;
28
29 # generate the parameters
30 ## generate alphas from U(0,1)
31 alphas = runif(I);
32 ## use the following betas
33 betas = c(rep(1,I/2), sample(c(0.5, 1.5), I/2,
       replace = TRUE));
34 ## generate gammas from N(0,1)
35 gammas = rnorm(S);
36
```

```
37
38 for(n in 1:N) {
39   # generate data with the given parameter set
40   data = gendata.M(gammas,alphas,betas);
41   # compress the data
42   mat = compress(data);
43
44   # write the dataset, full version
45   names.full[n] = paste(data.dir, str_pad(n, width=3,
         pad=0), "-full.data.csv", sep="");
46   write.csv(data, names.full[n], quote=FALSE, row.
       names=FALSE);
47   # write the dataset, compressed version
48   names.comp[n] = paste(data.dir, str_pad(n, width=3,
         pad=0), "-compressed.data.csv", sep="");
49   write.csv(mat, names.comp[n], quote=FALSE, row.
       names=FALSE);
50
51   # write the log file
52   cat("Dataset number", n, "\n", file=LOG_FILE,
       append=TRUE);
53   cat("Full matrix file:", names.full[n], "\n", file=
       LOG_FILE, append=TRUE);
54   cat("Compressed matrix file:", names.comp[n], "\n\n
       ", file=LOG_FILE, append=TRUE);
55   cat("----------------------------------------\n\n",
         file=LOG_FILE, append=TRUE);
56 }
57
58 if(print.param) {
59   cat("\nalphas = c(",paste(alphas,collapse=", "),")\
       n\n", file=LOG_FILE, append=TRUE);
60   cat("betas = c(",paste(betas,collapse=", "),")\n\n"
       , file=LOG_FILE, append=TRUE);
61   cat("gammas = c(",paste(gammas,collapse=", "),")\n\
       n", file=LOG_FILE, append=TRUE);
62 }
63
64 cat("\n\nFile generation completed.\n\n");
65 cat("\nFull matrix files:\nc(\"", paste(names.full,
     collapse="\", \""), "\")\n\n", sep="");
66 cat("\nCompressed matrix files:\nc(\"", paste(names.
     comp, collapse="\", \""), "\")\n\n", sep="");
```

Listing A.3: TMB template source code.

```
1 #include <TMB.hpp>
2 template<class Type>
3 Type objective_function<Type>::operator() () {
```

```
  /* data section */
  DATA_INTEGER(I);
  DATA_INTEGER(S);
  DATA_MATRIX(y);


  /* Parameter section */
  PARAMETER_VECTOR(alpha);
  PARAMETER_VECTOR(beta);
  PARAMETER_VECTOR(gamma);

  using namespace density;

  Type nll=0.0;      // Negative log likelihood
     function

  nll -= sum(dnorm(gamma,Type(0),Type(1),true));
     // gamma's ~ N(0,1)

  // nll from y
  for(int s=0;s<S;s++) {
    for(int i=0;i<I;i++)  {
      Type eta = alpha(i) + gamma(s)*(1+beta(i));
      Type prob =  exp(eta) / (1 + exp(eta));
      nll -= dbinom(y(s,i),Type(1),prob,true);
    }
  }

  return nll;
}
```

Listing A.4: AIC and BIC selection of the tuning parameter.

```
# +----------------------------------------+ #
# | > Simulation parameters                | #
# +----------------------------------------+ #
# string manipulation library
library(stringr);

full.files = paste("data/", list.files(path="./data/"
    , pattern="*full*"), sep="")[-(1:45)];

comp.files = paste("data/", list.files(path="./data/"
    , pattern="*compressed*"), sep="")[-(1:45)];

SIM = length(full.files); # number of random data-
    sets


```

```
14 # lambda sequence length
15 N = 200;
16 #lambda = c(0, exp(seq(from=0, to=5, length.out=N-1))
       );
17 lambda = seq(from=0, to=200, length.out=N);
18
19 # coefficients smaller than eps are considered 0
20 eps = 1e-12;
21
22 # simulation starting time
23 t0 = Sys.time();
24
25 library(TMB)
26 library(lbfgs)
27
28
29
30 # +----------------------------------------+ #
31 # | > Simulation                           | #
32 # +----------------------------------------+ #
33 pb = txtProgressBar(max=SIM*N, style=3);
34 for(sim in 1:SIM) {
35
36   # read the data
37   data = read.csv(full.files[sim], header=TRUE);
38   mat = read.csv(comp.files[sim], header=TRUE);
39
40   # number of subjects/items
41   S = NROW(data);
42   I = ncol(data);
43
44   # create f and df/d\theta
45   compile("MML.cpp")
46   dyn.load(dynlib("MML"))
47   ### MML part
48   parameters = list(alpha =rep(0,I), beta=rep(0,I),
         gamma = rep(0,S))
49   obj = MakeADFun(data=list(I=I, S=S,y=as.matrix(data
         )),parameters=parameters,
50                   DLL="MML",random=c("gamma"))
51   obj$env$tracemgc = FALSE
52   obj$env$inner.control$trace = FALSE
53   #obj$env$silent = TRUE
54
55   # prepare output structure
56   out = data.frame(lambda=lambda, p=NA, nll=NA, AIC=
         NA, BIC=NA);
57   # compute the zero-searching starting point
58   start = lbfgs(obj$fn, linesearch_algorithm="
```

```
      LBFGS_LINESEARCH_BACKTRACKING",
59                epsilon=10^-5,
60                obj$gr,
61                obj$par,
62                invisible=1)$par;
63  # compute AIC and BIC for each lambda value
64  for(i in 1:N) {
65    mle = lbfgs(obj$fn, linesearch_algorithm="
        LBFGS_LINESEARCH_BACKTRACKING",
66                epsilon=10^-5,
67                obj$gr,
68                start, #obj$par,
69                orthantwise_c=lambda[i],
70                orthantwise_start=I,
71                orthantwise_end=2*I,
72                invisible=1);
73    # print(mle);
74    out$p[i] = sum(abs(mle$par)>eps);
75    out$nll[i] = obj$fn(mle$par);#mle$value;
76    out$AIC[i] = 2 * (out$nll[i] + out$p[i]); #+ 2*p[
        i]*(p[i]+1)/(S*I-p[i]-1); # corrected AIC
77    out$BIC[i] = 2 * out$nll[i] + out$p[i] * log(S*I)
        ;
78
79    setTxtProgressBar(pb, (sim-1)*N+i);
80  }
81
82  # save:
83  ##  - number of non-zero parameters profile plot
84  svg(paste(c("./output",paste(str_pad(sim, width=3,
      pad=0), "-p.svg", sep=""),collapse="/"));
85  plot(out$lambda, out$p, type="o", pch=18, cex=0.7,
      main=paste(sim,"Number of non-zero parameters",
      sep=") "), xlab=expression(lambda), ylab="p");
86  dev.off();
87  ##  - negative log-likelihood profile plot
88  svg(paste(c("./output",paste(str_pad(sim, width=3,
      pad=0), "-nll.svg", sep=""),collapse="/"));
89  plot(out$lambda, out$nll, type="o", pch=18, cex
      =0.7,
90       main=paste(sim,"Negative penalized log-
           likelihood",sep=") "), xlab=expression(
           lambda), ylab="Negative Log-Likelihood");
91  dev.off();
92  ##  - negative AIC profile plot
93  svg(paste(c("./output",paste(str_pad(sim, width=3,
      pad=0), "-naic.svg", sep=""),collapse="/"));
94  plot(out$lambda, out$AIC, type="o", pch=18, cex
      =0.7, main=paste(sim,"Negative AIC",sep=") "),
```

```
      xlab=expression(lambda), ylab="AIC");
  dev.off();
  ##  - negative BIC profile plot
  svg(paste(c("./output",paste(str_pad(sim, width=3,
      pad=0), "-nbic.svg", sep="")),collapse="/"));
  plot(out$lambda, out$BIC, type="o", pch=18, cex
      =0.7, main=paste(sim,"Negative BIC",sep=") "),
      xlab=expression(lambda), ylab="BIC");
  dev.off();

  ## - the output
  write.csv(out, paste(c("./output",paste(str_pad(sim
      , width=3, pad=0),"-out.csv", sep="")),collapse=
      "/"), quote=FALSE, row.names=FALSE);


  cat("\n");

  dyn.unload(dynlib("MML"))
}
close(pb);
```

Listing A.5: Model estimation with maximum likelihood, AIC and BIC.

```
# +----------------------------------------+ #
# | > Summary elaboration                  | #
# +----------------------------------------+ #
# string manipulation library
library(stringr);
# input file list
full.files = paste("data/", list.files(path="./data/"
    , pattern="*full*"), sep="");
# comp.files = paste("data/", list.files(path="./data
    /", pattern="*compressed*"), sep="");
# output file list
out.files = paste("output/", list.files(path="./
    output/", pattern="???-out.csv"), sep="");
N = length(out.files);

# simulation parameters
##  sample size
S = 500;
I = 30;
##  true values:
###   generate alphas from U(0, 1)
alphas = c(0.687857406679541, 0.492192608769983,
    0.345115572912619, 0.995049911551178,
    0.695267170201987, 0.0107000356074423,
    0.345015853410587, 0.172049480024725,
```

```
    0.949360669590533, 0.249192638555542,
    0.732790308073163, 0.660289179766551,
    0.580316918902099, 0.594781526131555,
    0.866271492093801, 0.103902629809454,
    0.418307674117386, 0.867522824788466,
    0.352356912568212, 0.389825357589871,
    0.380464274203405, 0.642305639572442,
    0.521597083192319, 0.177710808347911,
    0.029990678653121, 0.773581623332575,
    0.486535674193874, 0.558618906186894,
    0.989597225794569, 0.701965618645772);
# set.seed(777);
# alphas = runif(I);
###   use the following betas
betas = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1.5, 1.5, 1.5, 0.5, 0.5, 0.5, 1.5, 0.5, 1.5,
    1.5, 0.5, 0.5, 1.5, 0.5, 0.5);
###  number of non-zero parameters
p = sum(betas!=1);

# output structure
##  best AIC chooice
aic = data.frame(sim=1:N, lambda.opt=NA, p=NA, nll=NA
    , AIC=NA);
aic[,paste("alpha",1:I,sep="")] = NA;
aic[,paste("beta",1:I,sep="")] = NA;
##  best BIC chooice
bic = data.frame(sim=1:N, lambda.opt=NA, p=NA, nll=NA
    , BIC=NA);
bic[,paste("alpha",1:I,sep="")] = NA;
bic[,paste("beta",1:I,sep="")] = NA;
##  MLE chooice
mle = data.frame(sim=1:N, p=NA, nll=NA);
mle[,paste("alpha",1:I,sep="")] = NA;
mle[,paste("beta",1:I,sep="")] = NA;
##  centered alphas
# alphas.centered = data.frame(MLE=rep(-alphas, I*N),
     AIC=rep(-alphas, I*N), BIC=rep(-alphas, I*N));
##  centered betas
# betas.centered = data.frame(MLE=rep(1-betas, I*N),
    AIC=rep(1-betas, I*N), BIC=rep(1-betas, I*N));


eps = 1e-6;
pb = txtProgressBar(max=N*4, style=3);
for(n in 1:N) {
  # read the n-th input
  data = read.csv(full.files[n]);
  # read the n-th output
```

```
52    out = read.csv(out.files[n]);
53    aic.opt = which.min(out$AIC);
54    bic.opt = which.min(out$BIC);
55
56    # create f and df/d\theta
57    compile("MML.cpp")
58    dyn.load(dynlib("MML"))
59    ### MML part
60    parameters = list(alpha =rep(0,I), beta=rep(0,I),
          gamma = rep(0,S))
61    obj = MakeADFun(data=list(I=I, S=S,y=as.matrix(data
          )),parameters=parameters,
62                    DLL="MML",random=c("gamma"))
63    obj$env$tracemgc = FALSE
64    obj$env$inner.control$trace = FALSE
65    #obj$env$silent = TRUE
66
67    # compute the zero-searching starting point
68    start = lbfgs(obj$fn, linesearch_algorithm="
          LBFGS_LINESEARCH_BACKTRACKING",
69                  epsilon=10^-5,
70                  obj$gr,
71                  obj$par,
72                  invisible=1)$par;
73    setTxtProgressBar(pb, 1+4*(n-1));
74
75    # MLE choice
76    hat = lbfgs(obj$fn, linesearch_algorithm="
          LBFGS_LINESEARCH_BACKTRACKING",
77                epsilon=10^-5,
78                obj$gr,
79                start, #obj$par,
80                orthantwise_c=0,
81                orthantwise_start=I,
82                orthantwise_end=2*I,
83                invisible=1);
84    mle[n,-(1:3)] = hat$par;
85    mle$p[n] = sum(abs(hat$par)>eps);
86    mle$nll[n] = obj$fn(hat$par);
87  #   alphas.centered$MLE[1:I+(n-1)*I] = alphas.
        centered$MLE[1:I+(n-1)*I] + unlist(mle[n,3+1:I]);
88  #   betas.centered$MLE[1:I+(n-1)*I] = betas.
        centered$MLE[1:I+(n-1)*I] + unlist(mle[n,3+I+1:I])
        ;
89    setTxtProgressBar(pb, 2+4*(n-1));
90    start = hat$par;
91
92    # AIC choice
93    hat = lbfgs(obj$fn, linesearch_algorithm="
```

```
      LBFGS_LINESEARCH_BACKTRACKING",
94              epsilon=10^-5,
95              obj$gr,
96              start, #obj$par,
97              orthantwise_c=out$lambda[aic.opt],
98              orthantwise_start=I,
99              orthantwise_end=2*I,
100             invisible=1);
101   aic[n,-(1:5)] = hat$par;
102   aic$lambda.opt = out$lambda[aic.opt];
103   aic$p[n] = sum(abs(hat$par)>eps);
104   aic$nll[n] = obj$fn(hat$par);
105   aic$AIC[n] = 2 * (aic$nll[n] + aic$p[n]);
106 #   alphas.centered$AIC[1:I+(n-1)*I] = alphas.
    centered$AIC[1:I+(n-1)*I] + unlist(aic[n,5+I+1:I])
    ;
107 #   betas.centered$AIC[1:I+(n-1)*I] = betas.
    centered$AIC[1:I+(n-1)*I] + unlist(aic[n,5+I+1:I])
    ;
108   setTxtProgressBar(pb, 3+4*(n-1));
109
110   # BIC choice
111   hat = lbfgs(obj$fn, linesearch_algorithm="
      LBFGS_LINESEARCH_BACKTRACKING",
112             epsilon=10^-5,
113             obj$gr,
114             start, #obj$par,
115             orthantwise_c=out$lambda[bic.opt],
116             orthantwise_start=I,
117             orthantwise_end=2*I,
118             invisible=1);
119   bic[n,-(1:5)] = hat$par;
120   bic$lambda.opt = out$lambda[bic.opt];
121   bic$p[n] = sum(abs(hat$par)>eps);
122   bic$nll[n] = obj$fn(hat$par);
123   bic$BIC[n] = 2 * bic$nll[n] + bic$p[n] * log(S*I);
124 #   alphas.centered$BIC[1:I+(n-1)*I] = alphas.
    centered$BIC[1:I+(n-1)*I] + unlist(bic[n,5+I+1:I])
    ;
125 #   betas.centered$BIC[1:I+(n-1)*I] = betas.
    centered$BIC[1:I+(n-1)*I] + unlist(bic[n,5+I+1:I])
    ;
126   setTxtProgressBar(pb, 4+4*(n-1));
127
128 }
129
130 save.image(file = "big-simulation-summary.RData");
```