lsirm12pl: An R Package for the Latent Space Item Response Model

Dongyoung Go, Gwanghee Kim, Jina Park, Junyong Park, Minjeong Jeon, Ick Hoon Jin

Abstract The item response model in latent space (LSIRM; Jeon et al. (2021)) uncovers unobserved interactions between respondents and items in the item response data by embedding both in a shared latent metric space. The R package lsirm12pl implements Bayesian estimation of the LSIRM and its extensions for various response types, base model specifications, and missing data handling. Furthermore, the lsirm12pl package provides methods to improve model utilization and interpretation, such as clustering item positions on an estimated interaction map. The package also offers convenient summary and plotting options to evaluate and process the estimated results. In this paper, we provide an overview of the LSIRM's methodological foundation and describe several extensions included in the package. We then demonstrate the use of the package with real data examples contained within it.

1 Introduction

Item response theory (IRT) models are a widely-used statistical approach to analyze assessment data in various fields, e.g., medical, educational, psychological, health, and marketing research (**??**). IRT models are designed to establish a relationship between observed item response data and unobserved person characteristics, commonly referred to as latent traits, e.g., competencies, attitudes, or personality (**??**). IRT models can predict the probability of a correct (or positive) response as a function of the respondents' latent traits and item features, such as item difficulty and discrimination. Additional technical details of IRT models are provided in the subsequent section.

Several R packages are available for estimating IRT models. The **ltm** package (**?**) is available to analyze dichotomous and polytomous item response data, including a one-parameter logistic (1PL) model (or the Rasch model), a two-parameter logistic (2PL) model, a three-parameter model (**??**) and a graded response model (**?**). The **eRm** package (**?**) estimates various extensions of the Rasch model, such as the rating scale model (RSM) (**?**), partial credit model (PCM) (**?**), linear logistic test model (LLTM) (**?**), the linear rating scale model (LRSM) (**?**), and the linear partial credit model (LPCM) (**??**). The **mirt** package (**?**) can estimate a wide range of IRT models, including exploratory and confirmatory multidimensional item response models. The **pcIRT** package (**?**) provides functions for estimating IRT models for polytomous (nominal) and continuous data, including the multidimensional polytomous Rasch model (**?**) and the continuous rating scale model (**?**), using conditional maximum likelihood (CML) estimation (**?**).

Conventional IRT models are typically based on two assumptions: conditional independence and homogeneity. Conditional independence assumes that item responses are independent of each other conditional on respondents' latent traits. The homogeneity assumption is, e.g., that respondents with the same ability have the same probability of answering a question correctly and that respondents have the same probability of answering a question with the same item difficulty. However, these assumptions are often violated in practice due to unobserved interactions between respondents and items, e.g., when particular items are more similar to each other (e.g., testlets) or when particular respondents show different probabilities of giving correct responses to certain items compared to other respondents with similar ability (e.g., differential item functioning). Violations of these assumptions can lead to biased parameter estimates and inferences (**???**). While some existing methods can address known violations prior to data analysis, there is currently no approach that enables the detection or management of unknown sources of violations in item response analysis, to the best of our knowledge.

? proposed a latent space item response model (LSIRM) that addresses such limitations of conventional IRT models. The LSIRM aims to estimate inherent interactions between respondents and items, alongside the latent traits of both. A key feature is its visual representation of these interactions in a low-dimensional latent space, called an *interaction map* in the form of distances between them. Interaction maps offer a clear and intuitive interpretation of complex respondent-item relationships, allowing users to identify patterns and clusters based on spatial proximity on the map. Additional details of the LSIRM are provided in a later section.

This paper presents the **lsirm12pl** package in R that offers Bayesian estimation of the LSIRM and its extensions. **?** focused on the response data for binary items and the Rasch model as the base model, and currently, no package is available to estimate the LSIRMs. To broaden the applicability of latent space item response modeling, **lsirm12pl** enables: (1) modeling continuous item responses; (2) missing data handling under different missing mechanisms assumptions (**?**); and (3) an extended base model specification using the 2PL model both for binary and continuous item response data. The package also offers options to cluster latent positions of items in the estimated interaction map using spectral clustering and the Neyman-Scott process model. The **lsirm12pl** supplies convenient summary and plotting options for interaction maps, model assessment, diagnosis, and result process and interpretation, aiming to improve the utilization of the LSIRM in practice.

The subsequent sections of the paper are structured as follows. To begin, we provide a concise overview of the 1PL and 2PL IRT models. We then delve into the LSIRM for binary response data and demonstrate how to fit the model with the package functions using a real dataset. Next, we extend the LSIRM to accommodate continuous data and provide guidance on fitting this extended model using the **lsirm12pl** package. In the end, we conclude the paper with final remarks and a discussion of future developments.

2 Item response theory models

IRT models are essential for analyzing item response data, which comprises respondents' answers to items on tests, surveys, or questionnaires. This section briefly discusses two widely utilized IRT models for dichotomous item response data.

2.1 1PL IRT Model

The 1PL model, also known as the Rasch model (**?**), is a classic IRT model for analyzing dichotomous item response data. Suppose $\boldsymbol{Y} = \{y_{k,i}\} \in \{0,1\}^{N\times P}$ is the $N \times P$ binary item response matrix under analysis, where $y_{k,i} = 1$ indicates a correct (or positive) response of the respondent k to the item i. In the Rasch model, the probability of the correct response for item i given by respondent k is given as follows:

$$\text{logit}(\mathbb{P}(y_{k,i} = 1|\theta_k, \beta_i)) = \theta_k + \beta_i, \;\; \theta_k \sim N(0, \sigma^2),$$

where $\theta_k \in \mathbb{R}$ is the respondent intercept parameter of respondent k, $\beta_i \in \mathbb{R}$ is the item intercept parameter of item i. The person intercept parameter represents the latent trait of interest, such as cognitive ability. The item intercept parameter represents the item easiness (or minus difficulty). As the respondent's ability increases, her/his likelihood of giving correct responses increases. On the other hand, the likelihood of correct responses decreases as the difficulty of the item increases. Note that based on the model, the probability of a respondent's giving a correct response to an item is a function of the two parameters – the person's ability and the item's difficulty. This means that respondents with the same level of ability are assumed to have the same probability of giving a correct response to an item. Similarly, items with the same level of difficulty are assumed to have the same probability of being correctly responded to. In other words, interactions between persons and items are not allowed in this model. However, in reality, respondents with the same level of ability

and items with the same level of difficulty may show a different success probability due to unobserved characteristics they may have, violating the assumptions.

2.2 2PL IRT Model

The 2PL model (**?**) extends the 1PL model by incorporating a discrimination parameter for each item. This parameter reflects the ability of the item to differentiate among respondents with similar abilities. The item discrimination parameters are also considered as item slopes, indicating how quickly the probability of correct responses increases as the respondent's abilities increase (**?**). A larger discrimination parameter indicates that the item is better at distinguishing between respondents with similar ability levels. In the 2PL model, the probability of person k giving a correct response to item i is given as follows:

$$\text{logit}(\mathbb{P}(y_{k,i} = 1|\theta_k, \alpha_i, \beta_i)) = \alpha_i\theta_k + \beta_i, \;\; \theta_k \sim N(0, \sigma^2), \tag{1}$$

where $\alpha_i \in \mathbb{R}$ is the discrimination parameter for the item i. For model identifiability, one of the item slope parameters is fixed at 1, e.g., $\alpha_1 = 1$. With the term $\alpha_i\theta_k$, the 2PL model allows for an interaction between respondents and items to some degree. However, it is not likely to capture all interactions between respondents and items that might not depend on respondents' ability (**?**).

3 Standard lsirm: 1pl lsirm for dichotomous data

The LSIRM (**?**) has been proposed as an extension of the conventional IRT models. The key idea of the LSIRM is to place respondents and items in a low-dimensional, shared metric space, called an interaction map, so that their unobserved interactions can be captured in the form of distances between them. Below we provide a brief overview of the LSIRM in its original form, presented for dichotomized data.

3.1 Statistical framework

To capture unobserved interactions between respondents and items, the original LSIRM (**?**) embeds items and respondents in an interaction map, i.e., a D-dimensional latent Euclidean space. Note that the interaction map is used as a tool to represent pairwise interactions between respondents and items; thus, the dimensions of an interaction map do not represent any specific quantity or have a substantive meaning. The original LSIRM is built on the 1PL IRT model; thus, we refer to the original LSIRM as the 1PL LSIRM.

The 1PL LSIRM assumes that the probability of giving a correct answer to item i by the respondent k is determined by a linear combination of the main effect of the respondent k, the main effect of the item i, and the pairwise distance between the latent position of item i and the latent position of respondent k. Then, the model is given by:

$$\text{logit}(\mathbb{P}(y_{k,i} = 1|\theta_k, \beta_i, \gamma, \boldsymbol{z_k}, \boldsymbol{w_i})) = \theta_k + \beta_i - \gamma d(\boldsymbol{z_k}, \boldsymbol{w_i}), \tag{2}$$

where $\theta_k \in \mathbb{R}$ and $\beta_i \in \mathbb{R}$ represent the respondent's latent trait and the item's easiness, respectively, same as in the conventional 1PL model. These two terms can also be seen as the main effects of respondents and items. The third term, $-\gamma d(\boldsymbol{z_k}, \boldsymbol{w_i})$, captures the interactions between respondents and items, where $\boldsymbol{z_k} \in \mathbb{R}^D$ and $\boldsymbol{w_i} \in \mathbb{R}^D$ are the latent position of the respondent k, and the latent position of the item i, respectively, where $\gamma \geq 0$ is the weight of the distance term $d(z_k, w_i)$. For a distance function $d : \mathbb{R}^D \times \mathbb{R}^D \mapsto [0, \infty)$, we use a Euclidean norm (that is, $d(z_k, w_i) = ||z_k - w_i||$) as a distance function in `lsirm1pl` to maintain simplicity and enhance the interpretability of the model (**?**). The weight of the distance term $\gamma \geq 0$ indicates the amount of interactions between respondents and items in the data, such that large γ implies stronger evidence for respondent by item interactions in the data. In contrast, near zero γ implies little evidence of respondent-item interactions

in the data, thus suggesting that one may go with the conventional IRT model without the interaction term.

The likelihood function of the observed data with the 1PL LSIRM is

$$\mathbb{L}\left(\boldsymbol{Y}=\boldsymbol{y}|\boldsymbol{\theta},\boldsymbol{\beta},\gamma,\boldsymbol{Z},\boldsymbol{W}\right)=\prod_{K=1}^{N}\prod_{i=1}^{P}\mathbb{P}(Y_{k,i}=y_{k,i}|\theta_k,\beta_i,\gamma,\boldsymbol{z_k},\boldsymbol{w_i}),$$

where $\boldsymbol{\theta}=(\theta_1,\ldots,\theta_N)$, $\boldsymbol{\beta}=(\beta_1,\ldots,\beta_P)$, $\boldsymbol{Z}=(\boldsymbol{z_1},\ldots,\boldsymbol{z_N})$ and $\boldsymbol{W}=(\boldsymbol{w_1},\ldots,\boldsymbol{w_P})$. Here, item responses are assumed to be independent conditional on the positions of respondents and items in an interaction map, as well as the main effects of respondent and item. This means that the traditional conditional independence assumption is alleviated with the model by accounting for respondent-item interactions. The detailed parameter estimations are described in the following Section.

3.2 Parameter Estimation

The R package **lsirm12pl** applies a fully Bayesian approach using the Markov chain Monte Carlo (MCMC) for estimation of the LSIRM. Following is the posterior distribution of the 1PL LSIRM.

$$\begin{aligned}\pi\left(\boldsymbol{\theta},\boldsymbol{\beta},\gamma,\boldsymbol{Z},\boldsymbol{W}|\boldsymbol{Y}=\boldsymbol{y}\right)\propto&\prod_{k=1}^{N}\prod_{i=1}^{P}\mathbb{P}(Y_{k,i}=y_{k,i}|\theta_k,\beta_i,\gamma,\boldsymbol{z_k},\boldsymbol{w_i})\\&\times\prod_{k=1}^{N}\pi(\theta_k)\times\prod_{i=1}^{P}\pi(\beta_i)\times\pi(\gamma)\times\prod_{k=1}^{N}\pi(\boldsymbol{z_k})\prod_{i=1}^{P}\pi(\boldsymbol{w_i})\end{aligned}$$

The priors for the model parameters are specified as follows:

$$\begin{aligned}\theta_k\mid\sigma^2&\sim N(0,\sigma^2),\ \sigma^2>0\\\beta_i\mid\tau_\beta^2&\sim N(0,\tau_\beta^2),\ \tau_\beta^2>0\\\log\gamma\mid\mu_\gamma,\tau_\gamma^2&\sim N(\mu_\gamma,\tau_\gamma^2),\ \mu_\gamma\in\mathbb{R},\ \tau_\gamma^2>0\\\sigma^2\mid a_\sigma,b_\sigma&\sim\text{Inv-Gamma}(a_\sigma,b_\sigma),a_\sigma>0,b_\sigma>0\\\boldsymbol{z_k}&\sim\text{MVN}_D(\boldsymbol{0},\boldsymbol{I_D})\\\boldsymbol{w_i}&\sim\text{MVN}_D(\boldsymbol{0},\boldsymbol{I_D}).\end{aligned}$$

where $\boldsymbol{0}$ is a D-vector of zeros and $\boldsymbol{I_D}$ is $D\times D$ identity matrix. The argument names and default values for the prior specifications in the **lsirm12pl** are described in our Github site [1].

To generate posterior samples for $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, $\boldsymbol{Z}$ and $\boldsymbol{W}$, we use the Metropolis-Hastings-within-Gibbs sampler (**?**). The conditional posterior distribution for each parameter is given in Equations in the Github site. We list the arguments, the default values for the jumping rules, and the standard deviations of the Gaussian proposal distributions in the Github site.

The log-odds of the probability of giving correct responses depend on the latent positions through distances, as discussed in **?**. Because distances are invariant to translations, reflections, and rotations of the positions of respondents and items, the likelihood function is invariant under these transformations. To resolve the identifiability issue of latent positions, the **lsirm12pl** package applies Procrustes transformation (**?**) as a post-processing of posterior samples, which is a standard procedure in the latent space modeling literature (**???**).

3.3 An Illustrated Example

Here, we demonstrate how to apply the 1PL LSIRM to real datasets using the **lsirm12pl** package. To this end, we use the Inductive Reasoning Developmental Test (TDRI) dataset (**?**) from the package, which contains item responses from 1,803 Brazilians (52.5% female)

[1] https://github.com/jiniuslab/lsirm12pl

of ages ranging from 5 to 85 years (M = 15.75; SD = 12.21). TDRI is a pencil-and-paper test consisting of 56 items that are designed to assess developmentally sequenced and hierarchically organized inductive reasoning.

The **lsirm12pl** package provides several functions for fitting the LSIRM, which requires setting hyperparameter values for prior distributions and/or tuning parameters for MCMC chains. By default, the `lsirm` function runs with the default settings unless otherwise specified by the user. The default MCMC run setting includes 15,000 iterations, 2,500 burn-ins, and 5 thinning. The base function for running the LSIRM is

```
lsirm(A ~ <term 1>(<term 2>,<term 3>,...))
```

where `A` is an item response matrix to be analyzed, `<term1>` is for the model option – either 'lsirm1pl' or 'lsirm2pl', while `<term 2>` and `<term 3>` are other specific options for the chosen model, which are detailed in the documentation of the **lsirm12pl** package. The following is an example of how to fit the 1PL LSIRM to the TDRI dataset (with no missing) with a default estimation setting with 4 MCMC chains using 2 multi-core processors. Additional details of other default values can be found in the documentation of the **lsirm12pl** package.

```
R > library("lsirm12pl")
R > data <- lsirm12pl::TDRI
R > data <- data[complete.cases(data),]
R > head(data)
  i1 i2 i3 i4 ... i54 i55 i56
1  1  1  1  1  ...  0   0   0
2  1  1  1  1  ...  0   0   0
3  1  1  1  1  ...  0   0   0
4  1  1  1  1  ...  0   0   0
5  1  1  1  1  ...  0   0   0

R > lsirm_result <- lsirm(data ~ lsirm1pl(chains = 4, multicore = 2, seed = 2025))
```

The estimation results of the model parameters $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, γ, $\boldsymbol{Z}$, and $\boldsymbol{W}$ are stored in individual lists per chain, while all results across the chains are stored in a single list.

The `summary()` function generates a summary of the first chain by default, but users can obtain summaries of other chains by setting the `chain.idx` option. The function provides posterior estimates of the "covariate coefficients" (model parameters such as β), allowing the users to select the `"mean"`, `"median"`, or `"mode"` through the `estimate` option. It also provides the highest posterior density interval (HPD) for the model parameters with the `CI` option that allows the users to set different significance levels. Additionally, the function supplies the Bayesian information criterion (BIC) and the maximum log posterior value. When the column names are available in the input data, the `summary()` function uses these names in the summary of the results.

```
R > summary(lsirm_result, chain.idx = 1, estimate = 'mean', CI = 0.95)
=========================
Summary of model
=========================

Call:         lsirm.formula(formula = data ~ lsirm1pl(chains = 4, multicore = 2, seed = 2025))
Model:         lpl LSIRM
Data type:         binary
Variable Selection:         FALSE
Missing:         NA
MCMC sample of size 15000, after burnin of 2500 iteration

Covariate coefficients posterior means of chain 1 :
```

```
    Estimate     2.5%   97.5%
i1   6.42354  5.82468  7.0642
...
i56  -1.01604 -2.58822  0.8133

--------------------------

Overall BIC (Smaller is better) : 43661.52

Maximum Log-posterior Iteration:
      value iter
[1,] -11487 137
```

The `diagnostic()` function checks the convergence of MCMC for each parameter using various diagnostic tools, such as trace plots, posterior density distributions, autocorrelation functions (ACF), and Gelman-Rubin-Brooks plots. The `diagnostic()` function has options: `draw.item`, and `gelman.diag`. The `draw.item` option in the `diagnostic()` function specifies the names and indexes of the parameters to diagnose. The `draw.item` option is set to a list where a key represents each parameter such as `` ``beta'',``theta'',``gamma'',``alpha'',``sigma'' ``, and `` ``sigma_sd'' ``, and the values indicate the indices of these parameters. The indexes can be expressed as vectors. In the following example code, the `draw.item` option is set as `list(``beta'' = c(1))` to check the diagnostic of the first index of the beta parameter, i.e. $\beta_1$. With `gelman.diag = TRUE`, the Gelman-Rubin convergence diagnostic, known as potential scale reduction factors (PSRF), is obtained.

```
R > diagnostic(lsirm_result,
           draw.item = list("beta" = c(1)),
           gelman.diag = TRUE)

Potential scale reduction factors:

          Point est. Upper C.I.
beta [i1]       1.01       1.04
```

Figure **??** displays the output of `diagnostic()` for β_1: trace plot (top left), posterior density plot (top right), autocorrelation plot (bottom left), and Gelman-Rubin-Brooks plot (bottom right). Different colors indicate different MCMC chains. Trace plots visualize the mixing of the MCMC chains. In a well-converged model, the trace plots should show that chains fluctuate consistently around a constant value, which indicates the parameter space is thoroughly explored. Density plots display the distribution of the sampled parameter values. In a converged model, the density plots should exhibit smooth, overlapping curves across different chains, which demonstrates that the samples are drawn from the same posterior distribution. Autocorrelation plots measure the correlation between samples at different lags. For a converged model, the autocorrelation should decrease rapidly as the lag increases, which suggests the samples are independent and the parameter space has been effectively explored. Gelman-Rubin-Brooks plots show a shrink factor, known as the potential scale reduction, which compares the variance within each chain to the variance between chains. A shrink factor value close to 1 indicates the within-chain variance is similar to the between-chain variance, providing evidence of good convergence. By examining these diagnostics collectively, the convergence of the model parameters from the LSIRM can be ensured.

The **lsirm12pl** package supplies the `gof()` function that assesses the goodness of fit of the LSIRM to the item response data being analyzed. A main diagnostic tool is a boxplot that displays the average posterior 'predicted' response value for each item (item-wise means)

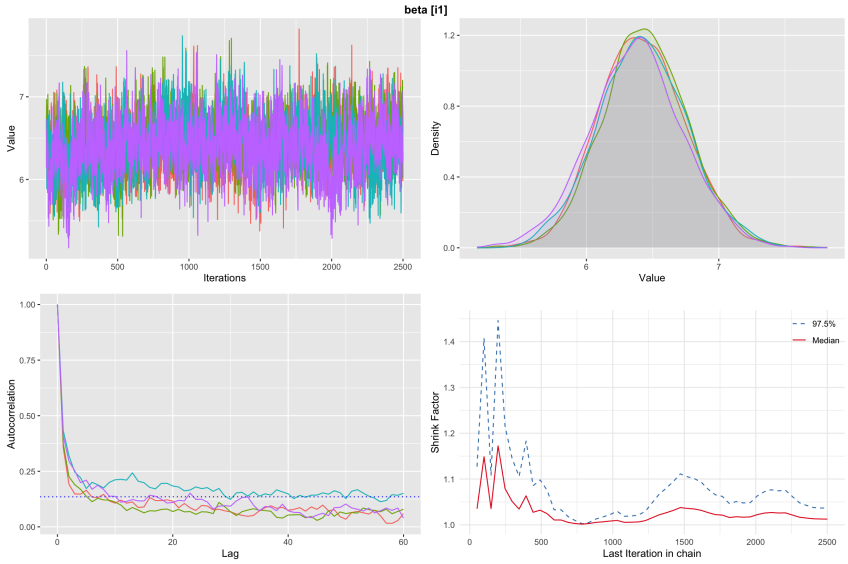


Figure 1: Diagnostics of β_1 using the `diagnostic()` function on the results of the 1PL LSIRM fitted to the TDRI dataset. The top left is the trace plot, the top right is the posterior density plot, the bottom left is the autocorrelation plot, and the bottom right is the Gelman-Rubin-Brooks plot. Different colors represent different MCMC chains.

with the average 'observed' response value for each item indicated by red dots. When the model fits well, the red dot is close to the midline of the boxplot. The `gof()` function includes a `chain.idx` option to choose a specific chain for assessing the goodness of fit. By default, the first MCMC chain is selected (i.e. `chain.idx=1`). When the `diagnostic()` function shows good convergence for all parameters, it does not matter which chain is chosen because the parameter estimates must be consistent across all chains. For illustration, we use the first MCMC chain by setting `chain.idx=1`.

For binary data, the `gof()` function additionally offers a receiver operating characteristic (ROC) curve. The ROC curve is a graphical representation that assesses the performance of a binary classifier. The area under the curve (AUC) quantifies the overall ability of the model to distinguish between classes, with values ranging from 0.5 to 1. A higher AUC and a curve close to the top-left corner indicate better performance.

```
R > gof(lsirm_result, chain.idx = 1)
```

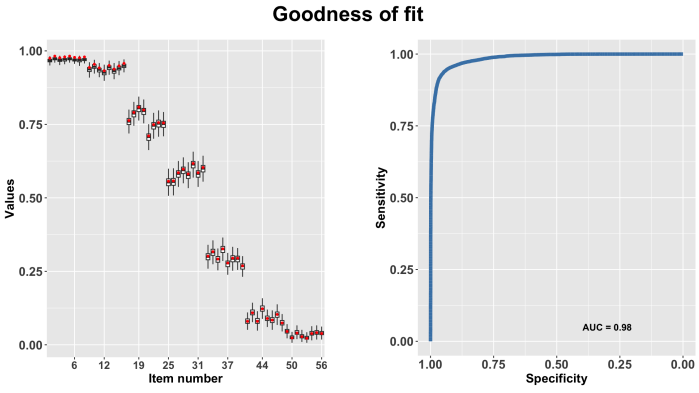
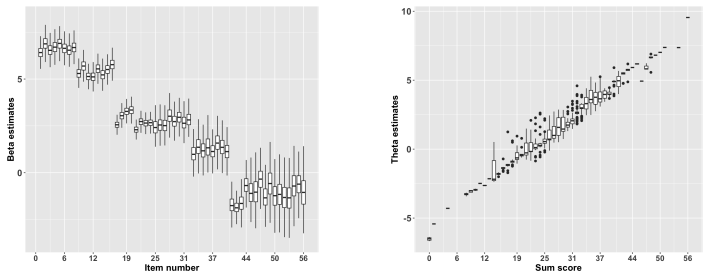


Figure 2: Goodness of fit of the 1PL LSIRM for the TDRI data using the `gof()` function. The box plots of the average posterior predicted response values for items (item-wise means) with the average observed response value for each item indicated by red dots (left) and the receiver operating characteristic (ROC) curve (right).

Figure **??** presents the output of the `gof()` function for the TDRI dataset, showing the results for the first MCMC chain, including both the boxplot (left) and the ROC curve (right). In this example, all of the red dots are closely located at the midline of the boxplot and the area under the curve (AUC) of the ROC curve approaches 1, suggesting that the fit of the LSIRM to the data is satisfactory.

The main outcome of the LSIRM fitting is the estimates of the respondent and item main effects and the interaction map. For a visual summary of these outputs, the user can use the `plot()` function with the `option` argument.

```
R > plot(lsirm_result, option = "beta", chain.idx = 1)
R > plot(lsirm_result, option = "theta", chain.idx = 1)
```



(a) β_i **(b)** θ_k

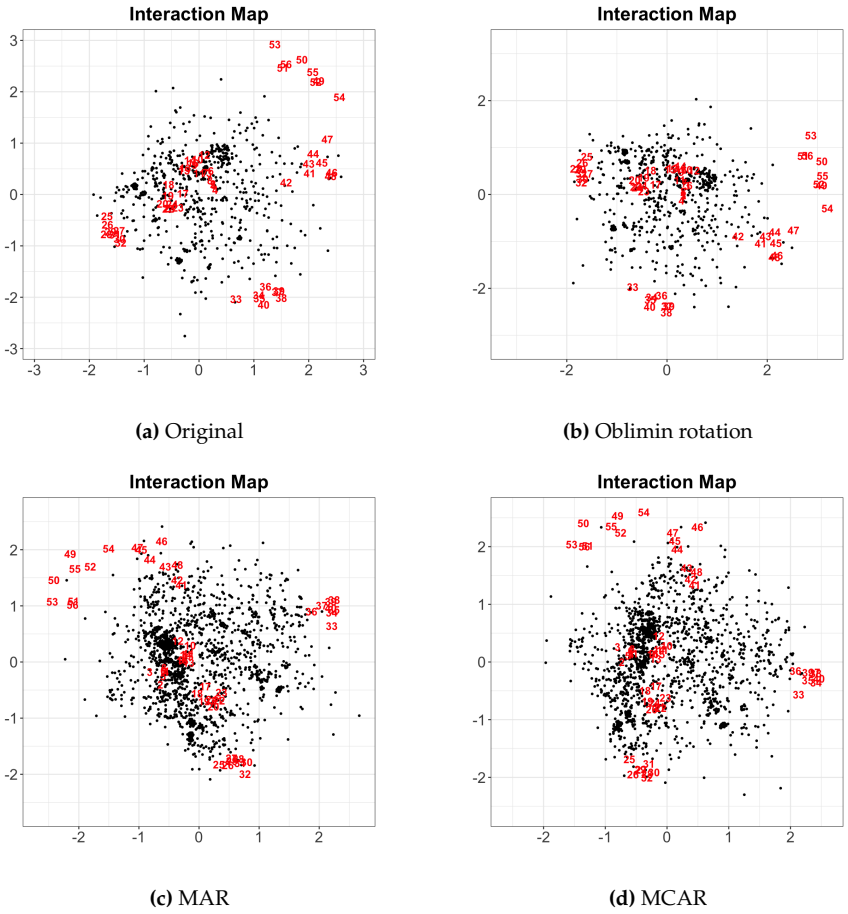
Figure 3: Visual summaries of the parameter estimate of β_i and θ_k using the `plot()` function based on the 1PL LSIRM fitted to the TDRI dataset. (a) Boxplots of the posterior samples for β_i; outliers are suppressed in the boxplots for the sake of simplicity. (b) Boxplots of the point estimate for θ_k as a function of the total sum scores of positive responses.

Figures **??** display the `plot()` results for summarizing β_i and θ_k by setting the `option` argument as the `"beta"` and `"theta"`, respectively. `option="beta"` generates the boxplots of the posterior samples for β_i, while `option="theta"` generates the boxplots of the point estimates for θ_k, plotted against the total sum scores of positive responses (ranging from 0 to P, where P is the number of items). In Figure **??**, as the item number increases (from left to right on the x axis), the β_k estimates decrease, indicating that earlier items are easier. In Figure **??**, individuals who correctly answer more items (i.e., higher sum scores) have higher θ_i estimates. It is sensible that those who answer more items correctly have higher θ_i values, as θ_i represents their ability levels.

The `plot()` also returns the interaction map with `option="interaction"`. The interaction map is created based on the estimated latent position of the item and respondent, $\boldsymbol{w}_i$ and $\boldsymbol{z}_k$, respectively. Note that the primary and unique advantage of the LSIRM lies in deriving intuitive information from the interaction map, based on the distances between respondents and items, between respondents, and between items. In the interaction map, a shorter distance between the latent position of the item i ($\boldsymbol{w_i}$) and the latent position of the respondent k ($\boldsymbol{z_k}$) indicates a stronger dependence (or interactions), which implies that the respondent k is more likely to respond correctly (or positively) to the item i, given the person's ability. In the **lsirm12pl** package, the interaction map is set to a two-dimensional space, as in **?**, for parsimony and interpretability.

```
R > plot(lsirm_result, option = "interaction", chain.idx = 1)
```

Figure **??** (top left) displays the interaction map based on the 1PL LSIRM for the TDRI data. The latent positions of items are represented as red numbers and respondents as black dots on the map. Note that the two coordinates (dimensions) of the map do not represent specific quantities or substantive meaning, as the interaction map is simply a tool to represent respondent-by-item interactions (**?**). In Figure **??**, we observe that respondents are widely spread around the center of the interaction map, while the items are separated by some clusters. For example, items located in the far north, such as items 50, 52, 53, 54,



(a) Original **(b)** Oblimin rotation

(c) MAR **(d)** MCAR

Figure 4: The estimated interaction maps based on the 1PL LSIRM fitted to the TDRI dataset. (a) The original interaction map. (b) The rotated interaction map using oblimin rotation. (c) The interaction map based on the MAR assumption. (d) The interaction map based on the MCAR assumption which has been reflected across the y-axis to facilitate comparisons with the other interaction maps. Red numbers represent item positions and black dots represent respondent positions in all plots.

and 55 (red numbers), are far from most respondents (black dots) on the map. This suggests that most respondents are less likely to respond correctly to those specific items, regardless of their ability levels. In other words, those items located in the south are difficult items, which is also confirmed in Figure **??**(a).

```
R > plot(lsirm_result, option = "interaction", rotation = TRUE, chain.idx = 1)
```

If desired, one can rotate the interaction map to improve the interpretability of the coordinates of the map with `rotation=TRUE`.

The **lsirm12pl** package offers the oblimin rotation (**?**) using the **GPArotation** package in R (**?**). Figure **??** (top left) shows the rotated version of the original map shown in Figure **??** (top right). In this example, the original and rotated maps appear pretty similar, although on the right there seems to be slight clockwise rotation compared to the one on the left; after rotation, the x- and y- coordinates may be interpreted based on the items that are closely placed to the respective coordinates.

Further, the **lsirm12pl** package offers an option to cluster latent positions of items. Two types of clustering methods are available: spectral clustering (**??**) and the Neyman-Scott process modeling approach (**???**), with `cluster` option as `spectral` and `neyman`, respectively. Spectral clustering is a technique that clusters points based on the eigenvalues of a similarity matrix, using the spectral properties of the data to identify clusters. The implementation of spectral clustering is based on the **kernlab** package (**?**) in R where the number of clusters

is determined using the average silhouette width (**?**). The Neyman-Scott point process modeling approach is a method to cluster points in time or space. Parent points (cluster center) are generated using a Poisson process, with offspring points clustered around each parent based on a defined probability distribution. The **lsirm12pl** package implements this method directly; it applies the MCMC algorithm a number of times independently to determine the distribution of the cluster number and cluster centers. Then, the mode of the cluster number distribution is chosen as the optimal cluster number. With the optimal cluster number, the cluster center that minimizes the Bayesian information criterion is selected, and items are assigned to the nearest cluster based on Euclidean distances.

```
R > plot(lsirm_result, cluster = "spectral", chain.idx = 1)

Clustering result (Spectral Clustering):
  group                                      item
      A              49, 50, 51, 52, 53, 54, 55, 56
      B              33, 34, 35, 36, 37, 38, 39, 40
      C              41, 42, 43, 44, 45, 46, 47, 48
      D              1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
                     11, 12, 13, 14, 15, 16, 17, 18,
                     19, 20, 21, 22, 23, 24
      E              25, 26, 27, 28, 29, 30, 31, 32

R > plot(lsirm_result, cluster = "neyman", chain.idx = 1)

|================================================| 100%
Clustering result (Neyman-Scott process):
  group                                      item
      A             33, 34, 35, 36, 37, 38, 39, 40
      B             49, 52, 54, 55
      C             25, 26, 27, 28, 29, 30, 31, 32
      D             17, 18, 19, 20, 21, 22, 23, 24
      E             50, 51, 53, 56
      F             9, 10, 11, 12, 13, 14, 15, 16
      G             1, 2, 3, 4, 5, 6, 7, 8
      H             41, 42, 43, 44, 45, 46, 47, 48
```

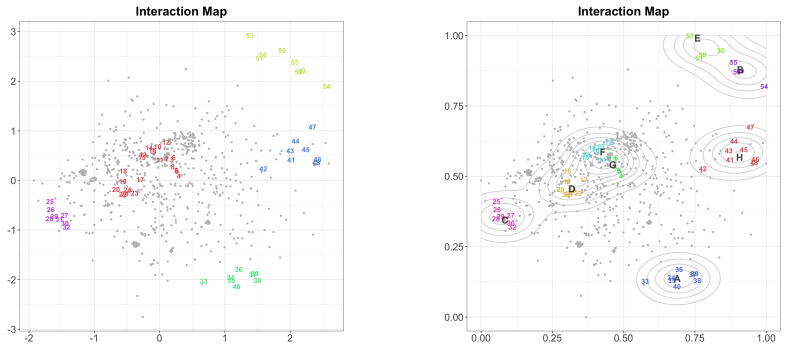
Figures **??** and **??** are item clustering results based on spectral clustering and the Neyman-Scott process model, respectively. In both plots, the gray dots indicate respondents, where numbers in colors indicate items with different cluster memberships. The Neyman-Scott process modeling approach additionally displays the center of the item cluster (alphabets A to H) and a contour for each item cluster. In this example, the spectral clustering method identifies five clusters, while the Neyman-Scott process modeling approach identifies eight clusters. Overall, the Neyman-Scott process modeling approach further split the items near the center (red items on the left) and in the north (light green items on the left) compared to the spectral clustering method.

3.4 Flexible Modeling Options

The **lsirm12pl** package provides a range of flexible modeling options for the LSIRM.

First, with `fixed_gamma` option, one can fix the distance weight γ to a constant value of 1. By doing so, the scale of the latent space can be standardized, easing comparison between different interaction maps. The default value of `fixed_gamma` option is `FALSE` (i.e., γ is freely estimated).

```
R > lsirm_result <- lsirm(data ~ lsirm1pl(fixed_gamma = TRUE))
```



(a) Spectral Clustering **(b)** Neyman-Scott Process Model

Figure 5: The interaction map with item clustering results based on the 1PL LSIRM fitted to the TDRI dataset, using (a) spectral clustering and the (b) Neyman-Scott process modeling approach. In both plots, the gray dots indicate respondents, while numbers in colors indicate items with different cluster memberships. The Neyman-Scott process approach additionally displays the center of the cluster (alphabets A to H) and a contour for each cluster.

Second, with `spikenslab` option, one can apply a model selection with the spike-and-slab prior for the distance weight γ (**?**). The spike-and-slab prior is a mixture of two log-normal priors: one is densely concentrated near zero, while another is more broadly spread across positive values. With this option, one can determine whether γ is zero or not, which in turn determines whether the distance term is needed for the data being analyzed. If γ is not zero, it is evidence for item-by-respondent interactions in the data being analyzed, and thus it would be useful to investigate the interactions between respondents and items with the LSIRM approach. The default value of `spikenslab` option is `FALSE`.

The posterior probability of γ being non-zero is indicated by the return value of `pi_estimate`. A `pi_estimate` value greater than 0.5 suggests that γ is likely non zero. Note that the two options `fixed_gamma` and `spikenslab` cannot be used simultaneously.

```
R > lsirm_result <- lsirm(data ~ lsirm1pl(spikenslab = TRUE))
R > lsirm_result$pi_estimate
[1] 0.9984
```

Third, the package offers options for handling missing data. In the context of missing data, three key assumptions are considered (**?**): missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR). MCAR occurs when the probability of missing data on a variable is unrelated to any other measured variables or the variable itself, making the missingness entirely random. MAR happens when the probability of missing data on a variable is related to some of the observed data but not the missing data itself, meaning the missingness can be explained by other observed variables. Unlike MCAR and MAR, MNAR assumes that the missing data mechanism depends on the unobserved data itself, making it challenging to estimate the model without strong assumptions or additional information about the missing data. Therefore, we focus on two types of missing data, MCAR and MAR, in the **lsirm12pl** package with the `missing_data` option.

With `missing_data = "mcar"`, the missing data are assumed to follow MCAR and the parameters are estimated solely based on the observed elements of the dataset being analyzed. On the other hand, with `missing_data="mar"`, the missing data are assumed to be MAR, and the data augmentation algorithm (**?**) is applied to impute missing values. With `missing_data="mar"`, the function returns the posterior samples of the imputed responses with `imp` and the probability of a correct response with `imp_estimate`. Imputed values are listed in the order of respondents. Note that all missing values should be recoded via `missing.val`. The percentage of missing data in the TDRI dataset is 30%, and missing values are replaced with 99 which is the default coding of missing data. The `missing_data` option

can be used in combination with other options such as (`spikenslab = TRUE`,`missing_data = "mar"`).

```
R > data <- lsirm12pl::TDRI
R > data[is.na(data)] <- 99
R > lsirm_result <- lsirm(data ~ lsirm1pl(missing_data = "mcar"))
R > lsirm_result <- lsirm(data ~ lsirm1pl(missing_data = "mar"))
R > lsirm_result$imp_estimate
[1] 0.9900 0.9868 0.9768 0.9884 0.9716 0.9716
...
[997] 0.9860 0.9788 0.8240 0.9924
 [ reached getOption("max.print") -- omitted 32327 entries ]
R > plot(lsirm_result, option = "interaction")
```

Figures **??**(c) and **??**(d) show the resulting interaction maps with MAR and MCAR assumptions, respectively. Note that a reflection is applied to the interaction map with MCAR assumptions (c) across the y-axis to ease comparisons with the other interaction plots. Reflection or rotation does not change the interpretations of the interaction map as interpretations are based on the distances, not the positions themselves. If the interaction maps of two missing assumptions are considerably different, further investigation is necessary to determine which assumption would be more suitable for the analyzed data. In this example, the interaction maps with the missing data options are similar to the original map presented in Figure **??**. Note that the original interaction map (a) is based on the complete item response data with no missing values; that is, the data includes only respondents who answered all items. In contrast, the interaction maps with the missing data option are based on MAR and MCAR assumptions. The similarity between these maps suggests that the observed only or imputed item response data provides a reasonable representation of the original item response data.

4 2pl lsirm for dichotomous data

The two-parameter LSIRM (2PL LSIRM) extends the 1PL LSIRM with item discrimination parameters.

4.1 Statistical framework

The 2PL LSIRMs the probability of a correct response by respondent k to item i as follows:

$$\text{logit}\Big(\mathbb{P}(Y_{k,i}=1|\theta_k,\alpha_i,\beta_i,\gamma,\boldsymbol{z_k},\boldsymbol{w_i})\Big)=\alpha_i\theta_k+\beta_i-\gamma d(\boldsymbol{z_k},\boldsymbol{w_i}),\;\; \theta_k\sim N(0,\sigma^2),$$

where θ_k, β_i, $\mathbf{z}_k$, $\mathbf{w}_i$ and γ have similar interpretations to Equation (**??**), while α_i represents the item discrimination parameters and one of the α_i parameters is fixed at 1, e.g., $\alpha_1 = 1$ to ensure identifiability.

The observed data likelihood function under the 2PL LSIRM is given as

$$\mathbb{L}\left(\boldsymbol{Y}=\boldsymbol{y}|\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{Z},\boldsymbol{W}\right)=\prod_{k=1}^{N}\prod_{i=1}^{P}\mathbb{P}(Y_{k,i}=y_{k,i}|\theta_k,\alpha_i,\beta_i,\gamma,\boldsymbol{z_k},\boldsymbol{w_i}).$$

4.2 Parameter Estimation

Following is the posterior distribution of the 2PL LSIRM.

$$\pi\left(\boldsymbol{\theta}, \boldsymbol{\beta}, \gamma, \boldsymbol{Z}, \boldsymbol{W} | \boldsymbol{Y} = \boldsymbol{y}\right) \propto \prod_{k=1}^{N}\prod_{i=1}^{P} \mathbb{P}(Y_{k,i} = y_{k,i} | \theta_k, \alpha_i, \beta_i, \gamma, \boldsymbol{z_k}, \boldsymbol{w_i}) \\ \times \prod_{k=1}^{N} \pi(\theta_k) \times \prod_{i=1}^{P} \pi(\alpha_i) \times \prod_{i=1}^{P} \pi(\beta_i) \times \pi(\gamma) \times \prod_{k=1}^{N} \pi(\boldsymbol{z_k}) \prod_{i=1}^{P} \pi(\boldsymbol{w_i})$$

The prior distributions for θ_k, β_i, $\mathbf{z}_k$, $\mathbf{w}_i$, and γ are identical for the 1PL LSIRM. For item discrimination parameters α, a log-normal distribution is used with a mean of μ_α and a variance of τ_α^2, as α is typically assumed to be positive. The argument names and default values for the prior specification are shown in the Github site. For the item discrimination parameters α, the arguments and default values are `pr_mean_alpha = 0.5,pr_sd_alpha = 1`.

The conditional posterior distribution for each parameter follows the same form as the Equation in the Github site. The jumping rule for each parameter is given in the Github site. The default jumping rule for α is `jump_alpha = 1`.

4.3 An Illustrated Example

We apply the 2PL LSIRM to the TDRI data. The default settings of the 2PL LSIRM are the same as the 1PL LSIRM except for α. The base function for the 2PL LSIRM is

```
lsirm(A ~ lsirm2pl())
```

The 2PL LSRIRM for dichotomous data was fitted with 4 MCMC chains using 2 multicore processors, as demonstrated in the following example code. Similarly to the `lsirm1pl` results, the estimation results for the model parameters $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, $\boldsymbol{\alpha}$, γ, $\boldsymbol{Z}$, and $\boldsymbol{W}$ for each chain are provided in individual lists.

```
R > library("lsirm12pl")
R > data <- lsirm12pl::TDRI
R > data <- data[complete.cases(data),]
R > lsirm_result <- lsirm(data ~ lsirm2pl(chains = 4, multicore = 2, seed = 2025))
```

To diagnose the results of the 2PL LSIRM analysis, we can use the `diagnostic()` function. Similarly to the diagnostic of the 1PL LSIRM, the convergence of MCMC for each parameter can be checked using various diagnostic tools, such as trace plots, posterior density distributions, autocorrelation functions (ACF), and Gelman-Rubin-Brooks plots. By setting the `draw.item` option to `list('beta'= c(1))`, we perform diagnostics for the β_i parameter of the first item ($i = 1$).

```
R > diagnostic(lsirm_result,
               draw.item = list(beta = c(1)),
               gelman.diag = T)

Potential scale reduction factors:

          Point est. Upper C.I.
beta [i1]          1          1
```

Figure **??** displays the diagnostic results for β_1 of the results of the 2PL LSIRM fitted to the TDRI dataset. These plots help us assess the convergence of MCMC for β_1. The interpretation of each plot from the `diagnostic()` function is the same as mentioned for the

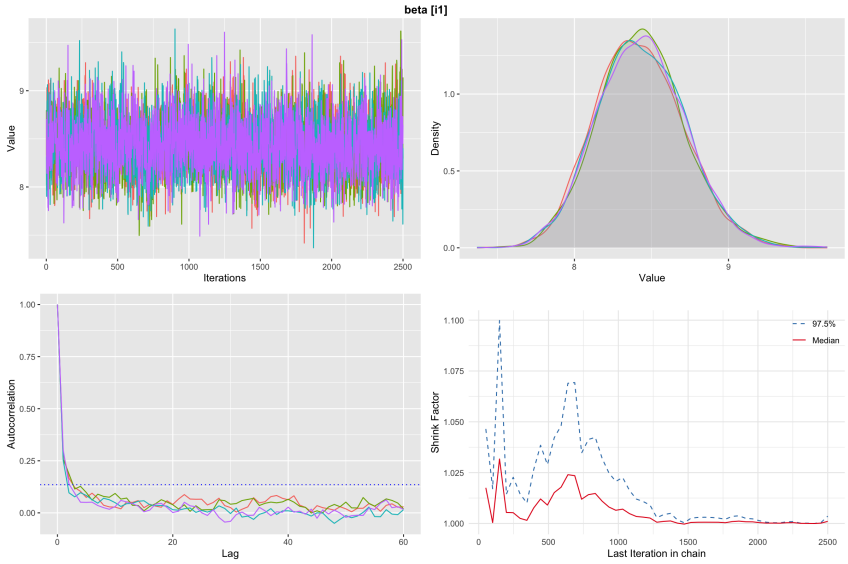


Figure 6: Diagnostics of β_1 using the `diagnostic()` function on the results of the 2PL LSIRM fitted to the TDRI dataset. The top left is the trace plot, the top right is the posterior density plot, the bottom left is the autocorrelation plot, and the bottom right is the Gelman-Rubin-Brooks plot. Different colors represent different MCMC chains.

1PL LSIRM in the previous Section. In Figure **??**, the convergence of β_1 was confirmed by various diagnostic tools.

The `gof()` function assesses the goodness-of-fit of the LSIRM. Figure **??** visualizes the box plots of the posterior predicted response values (item-wise means) compared with the observed item-wise means and the ROC curve to check the performance of the 2PL LSIRM fitted to the TDRI dataset. In the figure, most of the red dots are located close to the midline of the boxplots and the AUC is 0.97, indicating the fit of the 2PL LSIRM to the TDRI dataset is satisfactory.

```
R > gof(lsirm_result, chain.idx = 1)
```

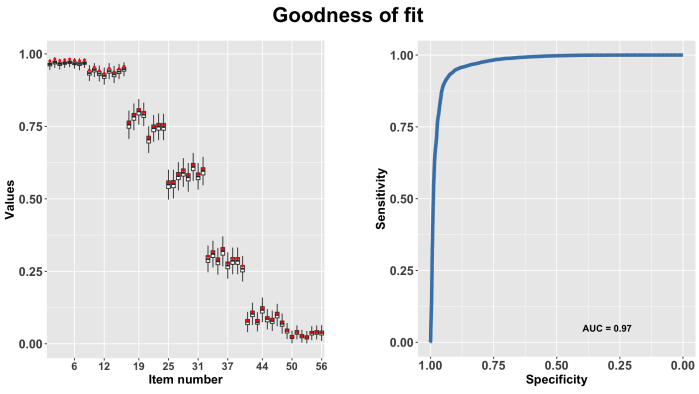
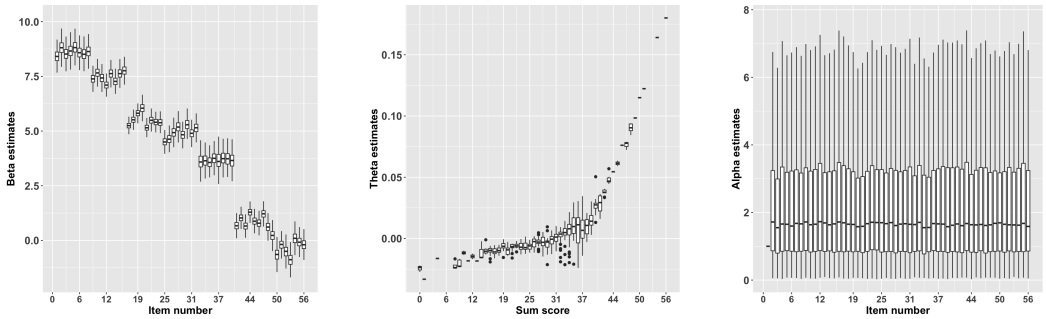


Figure 7: Goodness of fit of the 2PL LSIRM for the TDRI data using the `gof()` function. The box plots of the average posterior predicted response values for items against the average observed response value for each item indicated by red dots (left) and the receiver operating characteristic (ROC) curve (right).

Similarly to the 1PL LSIRM, the `plot()` function can generate different graphs for the 2PL LSIRM results with the `option` argument:

```
R > plot(lsirm_result, option = "beta", chain.idx = 1)
R > plot(lsirm_result, option = "theta", chain.idx = 1)
R > plot(lsirm_result, option = "alpha", chain.idx = 1)
```



(a) β_i **(b)** θ_k **(c)** α_i

Figure 8: Summarizing β_i and θ_k using the `plot()` function on the results of the 2PL LSIRM fitted to the TDRI dataset. (a) Boxplots of the posterior samples for β_i. (b) Boxplots of the point estimate for θ_k as a function of the total sum scores of positive responses. (c) Box plots of the posterior samples for α_i.

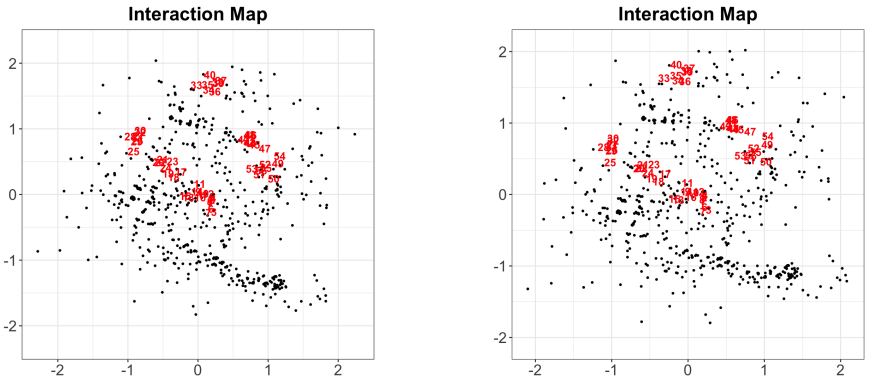
Figure **??** illustrates the results of the `plot()` function with the `"beta"`, `"theta"`, and `"alpha"` options for the 2PL LSIRM, respectively. Similarly to the 1PL LSIRM, β_i decreases for later items, indicating that later items are more difficult than earlier items. In addition, respondents with more correctly answered items (that is, higher sum scores) have higher θ_k estimates. Lastly, the distribution of the posterior samples for α_i is similar across all items.

The `plot()` function can be used to create a visualization of the interaction map based on the 2PL LSIRM by setting `option` argument as `"interaction"`. Figures **??** and **??** show the original and rotated interaction maps, respectively. In the interaction maps based on the 1PL and 2PL LSIRM, we notice that although the clustering of items is similar, the distances between item groups appear smaller in the interaction map of the 2PL LSIRM. This difference arises because the 2PL LSRIM takes item discrimination (α) into account, which explains some degree of item-by-person interactions. The interaction map with the oblimin rotation (b) shows a slight clockwise rotation compared to the original interaction map (a), similarly to the 1PL LSIRM's case.

```
R > plot(lsirm_result, option = "interaction", chain.idx = 1)
R > plot(lsirm_result, option = "interaction", rotation = TRUE, chain.idx = 1)
```

The `plot()` function visualizes the cluster of the latent positions of items by using spectral clustering and the Neyman-Scott process modeling approach, achieved by setting the `cluster` option to `spectral` and `neyman`, respectively.

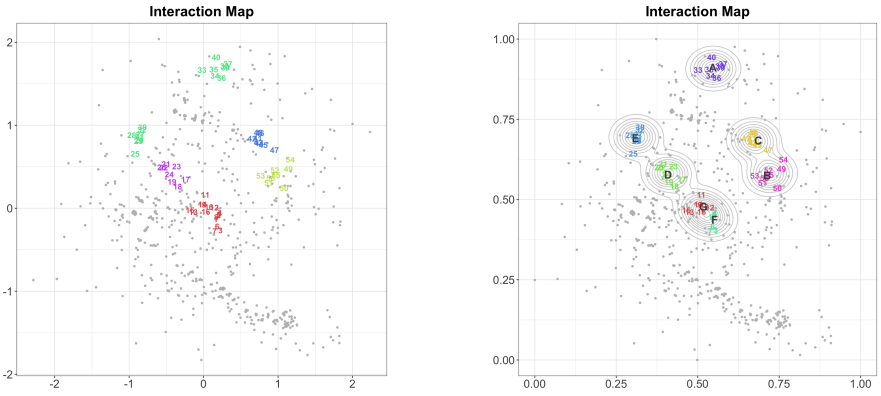
```
R > plot(lsirm_result, cluster = "spectral", chain.idx = 1)

Clustering result (Spectral Clustering):
  group                                item
      A         49, 50, 51, 52, 53, 54, 55, 56
      B         25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
                35, 36, 37, 38, 39, 40
      C         41, 42, 43, 44, 45, 46, 47, 48
      D         1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
                11, 12, 13, 14, 15, 16
      E         17, 18, 19, 20, 21, 22, 23, 24

 R > plot(lsirm_result, cluster = "neyman", chain.idx = 1)

|================================================| 100%
```



(a) Original **(b)** Oblimin rotation

Figure 9: The interaction map based on the 2PL LSIRM fitted to the TDRI dataset. (a) Visualization of the interaction map based on the 2PL LSIRM. (b) A rotated interaction map using oblimin rotation. Red numbers and black dots represent the latent positions for items and respondents, respectively, in both plots.

```
Clustering result (Neyman-Scott process):
  group                                item
      A         33, 34, 35, 36, 37, 38, 39, 40
      B         49, 50, 51, 52, 53, 54, 55, 56
      C         41, 42, 43, 44, 45, 46, 47, 48
      D         17, 18, 19, 20, 21, 22, 23, 24
      E         25, 26, 27, 28, 29, 30, 31, 32
      F         1, 2, 3, 4, 5, 6, 7, 8
      G         9, 10, 11, 12, 13, 14, 15, 16
```



(a) Spectral Clustering **(b)** Neyman-Scott Process Model

Figure 10: The interaction map with the item clustering results based on the 2PL LSIRM fitted to the TDRI dataset, using (a) spectral clustering and (b) the Neyman-Scott process modeling approach. In both plots, the gray dots indicate respondents, where numbers in colors indicate items with different cluster memberships. The Neyman-Scott process approach additionally displays the center of the cluster (alphabets) and a contour for each cluster.

Figure **??** and **??** display the result of spectral clustering and the Neyman-Scott process approach, respectively. The interpretation of gray dots, numbers, alphabets, and contours is the same as the 1PL LSIRM case. Spectral clustering resulted in 5 clusters, whereas the Neyman-Scott process approach resulted in 7 clusters. That is, the Neyman-Scott process approach identified more item groups than the spectral clustering. The overall clustering results are similar to the 1PL LSIRM case.

The flexible modeling options discussed with the 1PL LSIRM can be applied to the 2PL LSIRM. That is, `fixed_gamma` fixes the distance weight γ to 1 and `spikenslab` assigns a spike and slab prior to γ. Additionally, the two missing data options, `missing_data = "mcar"` and `missing_data = "mar"`, are available for the 2PL LSIRM.

```
R > lsirm_result <- lsirm(data ~ lsirm2pl(fixed_gamma = TRUE))
R > lsirm_result <- lsirm(data ~ lsirm2pl(spikenslab = TRUE))
R > lsirm_result <- lsirm(data ~ lsirm2pl(missing_data = "mcar"))
R > lsirm_result <- lsirm(data ~ lsirm2pl(missing_data = "mar"))
```

5 Lsirm for continuous item responses data

We consider an extension of the LSIRM for continuous item response data (LSIRM-continuous), which is done by using an appropriate link function following the generalized linear model framework (**?**).

5.1 Statistical framework

Consider the continuous item response data consisting of the $N \times P$ matrix $\boldsymbol{Y} = \{y_{k,i}\} \in \mathbb{R}^{N \times P}$. By choosing the identity link function for continuous item responses, the 1PL LSIRM-continuous is given as follows:

$$y_{k,i} \mid \boldsymbol{\theta}, \boldsymbol{\beta}, \gamma, \boldsymbol{Z}, \boldsymbol{W}, \sigma_\epsilon^2 = \theta_k + \beta_i - \gamma d(\boldsymbol{z_k}, \boldsymbol{w_i}) + \epsilon_{k,i}, \;\; \theta_k \sim \mathrm{N}(0, \sigma^2), \;\; \epsilon_{k,i} \sim \mathrm{N}(0, \sigma_\epsilon^2).$$

where θ_k and β_i are the main effects of the respondent k and the item i, respectively. $\boldsymbol{z_k}$ and $\boldsymbol{w_i}$ are the latent positions of the respondent k and the item i, respectively. An additional error term $\epsilon_{k,i} \sim N(0, \sigma_\epsilon^2)$ explains residuals unexplained by the 1PL LSIRM-continuous. A shorter distance between $\boldsymbol{w_i}$ and $\boldsymbol{z_k}$ indicates that the respondent k is likely to give a higher response value to item i given the main effects of the respondent and the item.

It is straightforward to extend the 1PL LSIRM-continuous to the 2PL version by adding item discrimination (or slope) parameters α_i. The 2PL LSIRM-continuous for $y_{k,i}$ is given as follows:

$$y_{k,i} \mid \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \gamma, \boldsymbol{Z}, \boldsymbol{W}, \sigma_\epsilon^2 = \alpha_i \theta_k + \beta_i - \gamma d(\boldsymbol{z_k}, \boldsymbol{w_i}) + \epsilon_{k,i}, \;\; \theta_k \sim \mathrm{N}(0, \sigma^2), \;\; \epsilon_{k,i} \sim \mathrm{N}(0, \sigma_\epsilon^2).$$

The interpretations of the model parameters in the 2PL LSIRM-continuous are similar to the case of the 1PL LSIRM-continuous and the 2PL LSIRM. For model identifiability, one of the item slopes is fixed to 1, e.g., $\alpha_1 = 1$.

The likelihood function of the 1PL LSIRM-continuous is given as

$$\begin{aligned} \mathbf{L}\left(\boldsymbol{Y} = \boldsymbol{y} | \boldsymbol{\theta}, \boldsymbol{\beta}, \gamma, \boldsymbol{Z}, \boldsymbol{W}, \sigma_\epsilon^2\right) &= \prod_{k=1}^{N} \prod_{i=1}^{P} \mathbf{L}(Y_{k,i} = y_{k,i} | \theta_k, \beta_i, \gamma, \boldsymbol{z_k}, \boldsymbol{w_i}, \sigma_\epsilon^2), \\ &= \prod_{k=1}^{N} \prod_{i=1}^{P} N(y_{k,i}; \theta_k + \beta_i - \gamma ||\boldsymbol{z_k} - \boldsymbol{w_i}||, \sigma_\epsilon^2), \end{aligned}$$

and the likelihood function of the 2PL LSIRM-continuous is given as

$$\mathbf{L}\left(\boldsymbol{Y} = \boldsymbol{y} | \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \gamma, \boldsymbol{Z}, \boldsymbol{W}, \sigma_\epsilon^2\right) = \prod_{k=1}^{N} \prod_{i=1}^{P} N(y_{j,i}; \alpha_i \theta_k + \beta_i - \gamma ||\boldsymbol{z_k} - \boldsymbol{w_i}||, \sigma_\epsilon^2).$$

5.2 Parameter Estimation

Following is the posterior distribution of LSRIM-continuous.

$$\pi\left(\boldsymbol{\theta}, \boldsymbol{\beta}, \gamma, \boldsymbol{Z}, \boldsymbol{W}, \sigma_\epsilon^2 | \boldsymbol{Y} = \boldsymbol{y}\right) \propto \prod_{k=1}^{N}\prod_{i=1}^{P} N(y_{k,i,}; \theta_k + \beta_i - \gamma||\boldsymbol{z_k} - \boldsymbol{w_i}||, \sigma_\epsilon^2) \times \prod_{k=1}^{N} \pi(\theta_k) \times \prod_{i=1}^{P} \pi(\beta_i) \times \pi(\gamma) \times \prod_{k=1}^{N} \pi(\boldsymbol{z_k}) \times \prod_{i=1}^{P} \pi(\boldsymbol{w_i}) \times \pi(\sigma_\epsilon^2)$$

The priors of the model parameters in the LSIRM-continuous are set the same as the priors for the LSIRM. For the error term σ_ϵ^2, we set the prior as follows:

$$\sigma_\epsilon^2 | a_{\sigma_\epsilon}, b_{\sigma_\epsilon} \sim \text{Inv-Gamma}(a_{\sigma_\epsilon}, b_{\sigma_\epsilon}),\ a_{\sigma_\epsilon} > 0,\ b_{\sigma_\epsilon} > 0.$$

The conditional posterior distributions of the LSIRM-continuous are similar to the LSIRM, which are given in the Github site. The jumping rule defaults remain the same as in the binary case (shown in the Github site).

5.3 An Illustrated Example

We demonstrate the application of the LSIRM-continuous using the Big Five Personality Test (FPT) dataset (**?**), which is included in the package. Data were collected from 1,015,342 respondents using an interactive online personality test from 2016 to 2018. The "Big-Five Factor Markers" from the international personality item pool were used in the test, which consists of 50 questions with response categories 1 = disagree, 3 = neutral, and 5 = agree based on a five-point Likert scale. Negatively worded items were reverse-coded. For illustration purposes, we randomly selected a sample of 3,000 respondents from the original data. We treated the ordinal item responses as continuous data, which is a common practice in applied research and is generally considered acceptable for other models, such as factor analysis.

The main fitting function for running the 1PL and 2PL LSIRM-continuous is identical to the 1PL and 2PL LSIRM for binary data. The function automatically identifies the data type and applies the appropriate models. Following is the code for data pre-processing and 1PL LSIRM-continuous fitting.

```
R > data <- lsirm12pl::BFPT
R > data[(data==0)|(data==6)] = NA
R > reverse <- c(2, 4, 6, 8, 10, 11, 13, 15, 16, 17,
                 18, 19, 20, 21, 23, 25, 27, 32, 34,
                 36, 42, 44, 46)
R > data[, reverse] <- 6 - data[, reverse]
R > data <- data[complete.cases(data),]
R > head(data)
    EXT1 EXT2 EXT3 ... OPN8 OPN9 OPN10
1    2    3    2   ...  2    4    4
2    1    3    3   ...  4    3    4
3    4    3    3   ...  2    4    4
5    1    4    3   ...  3    5    3
6    1    3    2   ...  3    5    3
8    3    5    4   ...  4    3    4

R > lsirm_result <- lsirm(data ~ lsirm1pl(niter = 25000, nburn = 5000, nthin = 10,
                                          jump_beta = 0.08, jump_theta = 0.3,
                                          jump_gamma  = 1.0,
                                          chains = 4, multicore = 2, seed = 2025))
```

To ensure convergence of the parameters, a different number of iterations and jumping rules of parameters are applied in this example. The estimated results for each chain are returned in the list containing estimated information on $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, γ, $\boldsymbol{Z}$, $\boldsymbol{W}$, σ, and σ_ϵ. The functions `summary()`, `diagnostics()`, and `gof()` described for the 1PL LSIRM can be applied similarly to obtain a summary, diagnosis, and goodness-of-fit results, respectively.

```
R > diagnostic(lsirm_result, draw.item = list(beta = c("AGR1")))
```

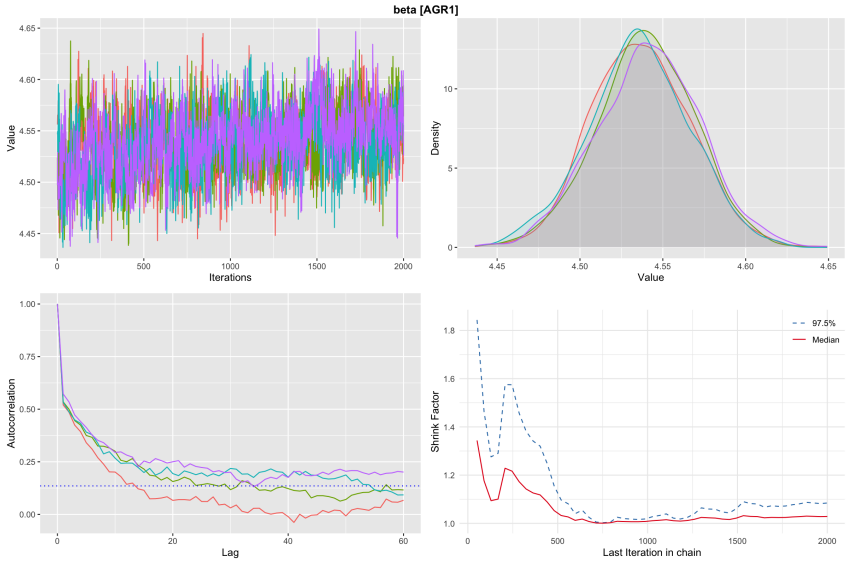


Figure 11: Diagnostics of β_{AGE_1} using the `diagnostic()` function on the results of the 1PL LSIRM-continuous fitted to the FPT dataset. The top left is the trace plot, the top right is the posterior density plot, the bottom left is the autocorrelation plot, and the bottom right is the Gelman-Rubin-Brooks plot. Different colors represent different MCMC chains.

Figure **??** displays the diagnostic results for β_{AGE_1} obtained using the `diagnostic()` function. The results suggest the convergence of β_{AGE_1} is achieved for this model.

Figure **??** shows the result of the goodness of fit assessment for the continuous example data. Unlike binary data, ROC is not available for continuous data, so the results of the `gof()` function include only the boxplots of the average predicted response values (item-wise means) against the observed item-wise means (marked with red dots). In this example, the red dots are located close to the midlines of the boxplots, implying a satisfactory model fit.

```
R > gof(lsirm_result, chain.idx = 1)
```

Same as before, the `plot()` function can be used to summarize the parameter estimate of β_i and θ_k.

```
R > plot(lsirm_result, option = "beta", chain.idx = 1)
R > plot(lsirm_result, option = "theta", chain.idx = 1)
```

Figure **??** shows the boxplots of the posterior samples for β_i. The parameter estimate of β_{35} is the smallest , while the estimates for β_{41} to β_{50} are relatively high. Figure **??** shows the boxplots of the point estimates of θ_k as a function of the sum scores of the observed responses binned into 10 groups to prevent overlap and improve readability on the x-axis. As expected, higher sum scores are aligned with higher θ_k values (indicating socially desirable characteristics).

Figure **??** illustrates the interaction map derived from the 1PL LSIRM-continuous. In the interaction map, the latent positions of the respondents are positioned around the center, while the items are scattered in a triangular pattern showing roughly three clusters (in the

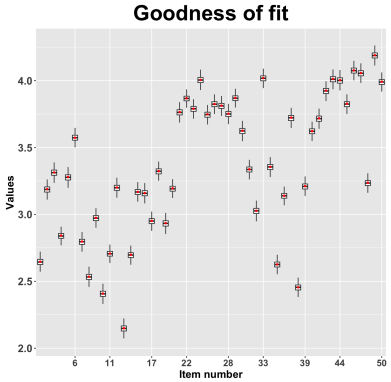
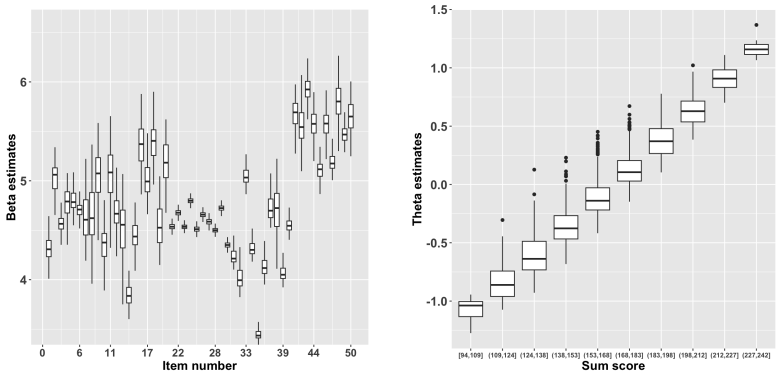


Figure 12: Goodness of fit of the 1PL LSIRM-continuous for the BFPT dataset using the `gof()` function. The box plots of the average posterior predicted response value for items against the average observed response value for each item indicated by red dots.



(a) β_i **(b)** θ_k

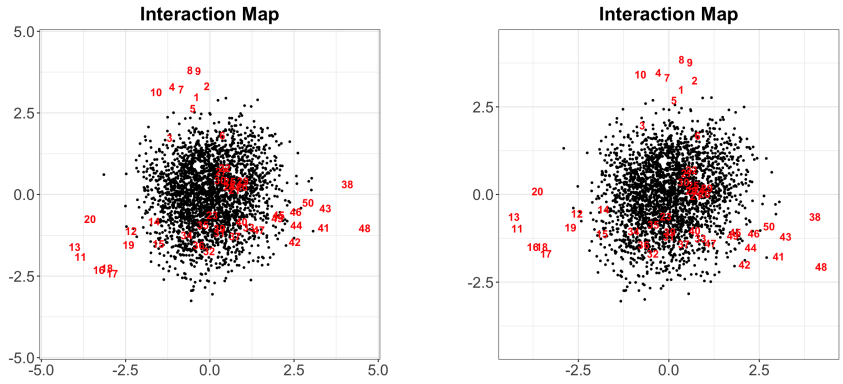
Figure 13: Visual summaries of β_i and θ_k using the `plot()` function on the 1PL LSIRM-continuous fitted to the FPT dataset. (a) Boxplots of the posterior samples for β_i. (b) Boxplots of the point estimates of θ_k as a function of the total sum scores of the responses binned into 10 groups

north, west, and east) in addition to the cluster around the center. The original interaction map is slightly rotated counterclockwise in the rotated interaction map, so that the items are more closely placed to the two coordinates.

```
R > plot(lsirm_result, chain.idx = 1)
R > plot(lsirm_result, rotation = TRUE, chain.idx = 1)
```

Figure **??** and Figure **??** depict the result of spectral clustering and the Neyman-Scott process modeling approach for the BFPT example dataset, respectively. Gray dots, numbers with colors, alphabets, and contours have the same interpretations as the clustering results presented earlier with the binary LSIRM models. In Figure **??**, the items in cluster C, highlighted in purple and located near the center of the interaction map, are closer to the latent positions of most people compared to other items. This implies that these items are more likely to receive higher response values. Conversely, items in clusters that are positioned farther from the center, such as clusters A, B, and E, are distant from many (but different groups of) respondents, indicating they are likely to receive lower response values by those who are far away from the corresponding item clusters.

```
R > plot(lsirm_result, cluster = "spectral", chain.idx = 1)

Clustering result (Spectral Clustering):
  group                                        item
```



(a) Original **(b)** Oblimin rotation

Figure 14: The interaction map based on the 1PL LSIRM-continuous fitted to the FPT dataset. (a) Visualization of the interaction map based on the 1PL LSIRM-continuous. (b) A rotated interaction map of the 1PL LSIRM-continuous using oblimin rotation. Red numbers and black dots represent the latent positions for items and respondents, respectively.

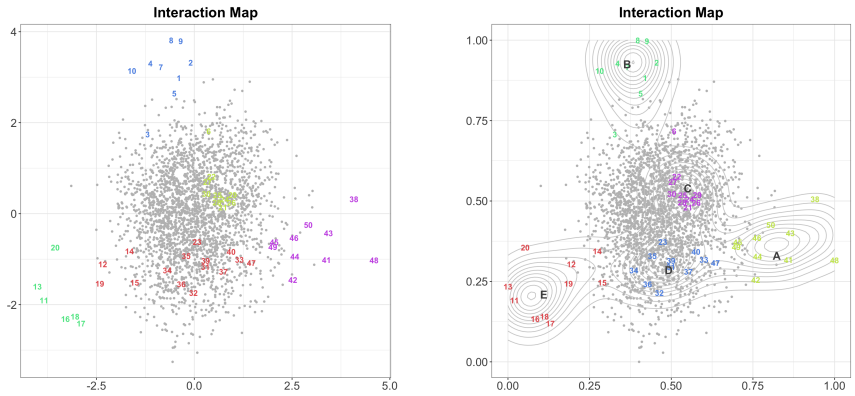
```
      A          11, 13, 16, 17, 18, 20
      B          38, 41, 42, 43, 44, 45, 46, 48, 49, 50
      C          1, 2, 3, 4, 5, 7, 8, 9, 10
      D          6, 21, 22, 24, 25, 26, 27, 28, 29, 30
      E          12, 14, 15, 19, 23, 31, 32, 33, 34, 35,
                 36, 37, 39, 40, 47

R > plot(lsirm_result, cluster = "neyman", chain.idx = 1)

|=================================================| 100%
Clustering result (Neyman-Scott process):
  group                                        item
      A          38, 41, 42, 43, 44, 45, 46, 48, 49, 50
      B          1, 2, 3, 4, 5, 7, 8, 9, 10
      C          6, 21, 22, 24, 25, 26, 27, 28, 29, 30
      D          23, 31, 32, 33, 34, 35, 36, 37, 39, 40, 47
      E          11, 12, 13, 14, 15, 16, 17, 18, 19, 20
```

The flexible modeling options discussed with the binary LSIRM can be applied to the functions for the LSIRM-continuous.

```
R > lsirm_result <- lsirm(data ~ lsirm1pl(fixed_gamma = TRUE))
R > lsirm_result <- lsirm(data ~ lsirm1pl(spikenslab = TRUE))
R > lsirm_result <- lsirm(data ~ lsirm1pl(missing_data = "mcar"))
R > lsirm_result <- lsirm(data ~ lsirm1pl(missing_data = "mar"))
```

6 Conclusion

In this paper, we introduced the R package **lsirm12pl** for estimating the LSIRM (**?**) and its extensions. The LSIRM is a powerful extension of conventional IRT models that allow for the estimation and visualization of potential interactions between respondents and items in an interaction map, a low dimensional Euclidean space. The original LSIRM framework was proposed for binary item response data based on the 1PL IRT base model. To broaden its applicability, we extended the original LSIRM to cover different response types and model specifications. Further, we added several useful options, e.g., for handling missing data, item clustering, and model assessment.



(a) Spectral Clustering **(b)** Neyman-Scott Process Model

Figure 15: The interaction map with item clustering results based on the 1PL LSIRM model-continuous fitted to the FPT dataset, using (a) spectral clustering and (b) Neyman-Scott process approach. In both plots, the gray dots indicate respondents, where numbers in colors indicate items with different cluster memberships. The Neyman-Scott process approach additionally displays the center of the cluster (alphabets) and a contour for each cluster.

It is worth noting that one may observe some variability in clustering results across multiple chains. Such variability stems from at least three factors: First, clustering is an unsupervised method, meaning that results are not uniquely determined and may naturally vary across Markov chains. Second, we apply the Neyman-Scott Process clustering to the posterior distributions of the LSIRM model, and the clustering outcomes are therefore directly influenced by the estimated LSIRM results. Third, Bayesian models inherently incorporate uncertainty, reflected in the posterior distributions and subsequently in the clustering outcomes. Having said that, we noted in our empirical analysis that the overall clustering structure was pretty stable across multiple chains. For critical applications, we recommend implementing a consensus clustering approach by aggregating results across multiple chains to identify the most robust and stable groupings.

A fully Bayesian approach was used for model estimation with a Metropolis-Hastings-within-Gibbs sampler. The **lsirm12pl** package offers default estimation settings for priors, jumping rules, number of iterations, burn-ins, and thinning. The default estimation setting works reasonably well in a broad range of situations, but users can manually revise the estimation settings if desired. In addition, the package **lsirm12pl** offers convenient supplemental functions to evaluate, summarize, visualize, diagnose, and interpret the estimated results. We provided detailed illustrations of the package with real data examples available in the package. We hope that these illustrations guide researchers in using the **lsirm12pl** package for the analysis of their own datasets.

The R package **lsirm12pl** is our first step in making the LSIRM approach more applicable and usable in practice. The code is written using **Rcpp** (**???**) and **RcppArmadillo** (**?**) in R for efficient computation. For example, in our first data example with 726 respondents and 56 test items, the computation time for one chain using a single core `lsirm1pl` was 1.93 minutes on an Apple M1 laptop. We provide additional details of our package implementation in the package GitHub site (https://github.com/jiniuslab/lsirm12pl). We will continue to update the package by incorporating additional modeling, data analysis, and visualization options to make the **lsirm12pl** package more useful in a wider range of situations. For example, we are currently engaged in the development of other extensions, such as for ordinal and longitudinal data. As advancements are made, we will systematically include them in the software package. This ongoing development will further enhance the utility and applicability of the LSIRM in various practical scenarios.

7 Acknowledgement

We thank the Editor, Associate Editor, and reviewers for their constructive comments. We also thank Dr. Won Chang for constructive comments on our work. This work was partially supported by the National Research Foundation of Korea [grant number NRF-2020R1A2C1A01009881, RS-2023-00217705, RS-2024-00333701; Basic Science Research Program awarded to IHJ], [grant number NRF-2025S1A5C3A0200632411; awarded to IHJ and MJ], and the ICAN (ICT Challenge and Advanced Network of HRD) support program [grant number RS-2023-00259934], supervised by the IITP (Institute of Information & Communications Technology Planning & Evaluation). Correspondence should be addressed to Ick Hoon Jin, Department of Applied Statistics, Department of Statistics and Data Science, Yonsei University, Seoul, Republic of Korea. Go, Kim, and Park are co-first authors and listed in alphabetical order.

Dongyoung Go
Department of Statistics and Data Science
Department of Applied Statistics
Yonsei University
50 Yonsei-ro, Seodaemun, Seoul, Republic of Korea
dongyoung.gr@yonsei.ac.kr

Gwanghee Kim
Department of Statistics and Data Science
Yonsei University
50 Yonsei-ro, Seodaemun, Seoul, Republic of Korea
musagh08@yonesi.ac.kr

Jina Park
Department of Statistics and Data Science
Department of Applied Statistics
Yonsei University
50 Yonsei-ro, Seodaemun, Seoul, Republic of Korea
pja070707@yonesi.ac.kr

Junyong Park
Samsung Electronics, Formerly at Yonsei University
1 Samsung Electronics-ro, Hwaseong-si, Gyeonggi-do, Republic of Korea
jun94.park@samsung.com

Minjeong Jeon
School of Education and Information Studies
University of California, Los Angeles
405 Hilgard Avenue, Los Angeles, CA 90095
mjjeon@ucla.edu

Ick Hoon Jin
Department of Statistics and Data Science
Department of Applied Statistics
Yonsei University
50 Yonsei-ro, Seodaemun, Seoul, Republic of Korea
ijin@yonsei.ac.kr