mudfold

Balafas, Spyros E.; Krijnen, Wim P.; Post, Wendy J.; Wit, Ernst C.

Published in:
The R Journal

Publication date:
2020

mudfold: An R Package for Nonparametric IRT Modelling of Unfolding Processes

by Spyros E. Balafas, Wim P. Krijnen, Wendy J. Post and Ernst C. Wit

Abstract Item response theory (IRT) models for unfolding processes use the responses of individuals to attitudinal tests or questionnaires in order to infer item and person parameters located on a latent continuum. Parametric models in this class use parametric functions to model the response process, which in practice can be restrictive. MUDFOLD (Multiple UniDimensional unFOLDing) can be used to obtain estimates of person and item ranks without imposing strict parametric assumptions on the item response functions (IRFs). This paper describes the implementation of the MUDFOLD method for binary preferential-choice data in the R package **mudfold**. The latter incorporates estimation, visualization, and simulation methods in order to provide R users with utilities for nonparametric analysis of attitudinal questionnaire data. After a brief introduction in IRT, we provide the methodological framework implemented in the package. A description of the available functions is followed by practical examples and suggestions on how this method can be used even outside the field of psychometrics.

Introduction

In this paper we introduce the R package **mudfold** (Balafas et al., 2019), which implements the nonparametric IRT model for unfolding processes MUDFOLD. The latter, was developed by Van Schuur (1984) and later extended by Post (1992) and Post and Snijders (1993). IRT models have been designed to measure mental properties, also called latent traits. These models have been used in the statistical analysis of categorical data obtained by the direct responses of individuals to tests and questionnaires. Two response processes that result in different classes of IRT models can be distinguished. The cumulative (also called monotone) processes and the unfolding (also called proximity) processes in the IRT framework differ in the way that they model the probability of a positive response to a question from a person as a function of the latent trait, which is termed as item response function (IRF).

Cumulative IRT models also known as Rasch models (Rasch, 1961), assume that the IRF is a monotonically increasing function. That is, the higher the latent trait value for a person, the higher the probability of a positive response to an item (Sijtsma and Junker, 2006). This assumption makes cumulative models suitable for testing purposes where latent traits such as knowledge or abilities need to be measured. The unfolding models also known as proximity models consider nonmonotone IRFs. These models originate from the work of Thurstone (1927, 1928) and have been formalized by Coombs (1964) in his deterministic unfolding model. In unfolding IRT the IRF is assumed to be a unimodal (single 'peak') function of the distance between the person and item locations on a hypothesized latent continuum. Unimodal IRFs imply that the closer an individual is located to an item the more likely is that he responds positively to this item (Hoijtink, 2005). Unfolding models can be used when one is interested to measure bipolar latent traits such as preferences, choices, or political ideology, which are generally termed as attitudes (Andrich, 1997). Such type of latent traits when they are analyzed using monotone IRT models usually result in a multidimensional solution. In this sense, unfolding models are more general than the cumulative IRT models (Stark et al., 2006; Chernyshenko et al., 2007) and can be seen as a form of quadratic factor analysis (Maraun and Rossi, 2001).

Parametric IRT (PIRT) models for unfolding processes exist for dichotomous items (Hoijtink, 1991; Andrich and Luo, 1993; Maydeu-Olivares et al., 2006), polytomous items (Roberts and Laughlin, 1996; Luo, 2001) as well as for bounded continuously scored items (Noel, 2014). Typically, estimation in PIRT models exploits maximum likelihood methods like the marginal likelihood (e.g. Roberts et al., 2000) or the joint likelihood (e.g. Luo et al., 1998), which are optimized using the expectation-maximization (EM) or Newton type of algorithms. Unfolding PIRT models that infer model parameters by adopting Bayesian Markov Chain Monte Carlo (MCMC) algorithms (Johnson and Junker, 2003; Roberts and Thompson, 2011; Liu and Wang, 2019; Lee et al., 2019) are also available. PIRT models however, make explicit parametric assumptions for the IRFs, which in practice can restrict measurement by eliminating items with different functional properties.

Nonparametric IRT (NIRT) models do not assume any parametric form for the IRFs but instead introduce order restrictions (Sijtsma, 2005). These models have been used to construct or evaluate scales that measure among others, internet gaming disorder (Finserås et al., 2019), pedal sensory loss (Rinkel et al., 2019), partisan political preferences (Hänggli, 2020), and relative exposure to soft

versus hard news (Boukes and Boomgaarden, 2015). The first NIRT model was proposed by Mokken (1971) for monotone processes. His ideas were used for the unfolding paradigm by Van Schuur (1984) who designed MUDFOLD as the unfolding variant of Mokken's model. MUDFOLD was extended by Van Schuur (1992) for polytomous items and Post (1992) and Post and Snijders (1993) derived testable properties for nonparametric unfolding models that were adopted in MUDFOLD. Usually, NIRT methods employ heuristic item selection algorithms that first rank the items on the latent scale and then use these ranks to estimate individual locations on the latent continuum. Such estimates for individuals' ideal-points in unfolding NIRT have been introduced by Van Schuur (1988) and later by Johnson (2006). NIRT approaches can be used for exploratory purposes, preliminary to PIRT models, or in cases where parametric functions do not fit the data.

IRT models can be fitted by means of psychometric software implemented in R (Choi and Asilkalkan, 2019), which can be downloaded from the Comprehensive R Archive Network (CRAN)[1]. An overview of the R packages suitable for IRT modelling can be found at the dedicated task view *Psychometrics*. PIRT models for unfolding where the latent trait is unidimensional, such as the graded unfolding model (GUM) (Roberts and Laughlin, 1996) and the generalized graded unfolding model (GGUM) (Roberts et al., 2000) can be fitted by the R package **GGUM** (Tendeiro and Castro-Alvarez, 2018). Sub-models in the GGUM class are also available into the Windows software **GGUM2004** (Roberts et al., 2006). A large variety of unfolding models for unidimensional and multidimensional latent traits can be defined and fitted to data with the R package **mirt** (Chalmers, 2012). To our knowledge, software that fits nonparametric IRT in the unfolding class of models (analogous to the **mokken** package (Van der Ark, 2007, 2012) in the cumulative class) is not yet available in R.

In order to fill this gap, we have developed the R package **mudfold**. The main function of the package implements item selection algorithm of Van Schuur (1984) for scaling the items on a unidimensional scale. Scale quality is assessed using several diagnostics such as, scalability coefficients similar to the homogeneity coefficients of Loevinger (1948), statistics proposed by Post (1992), and newly developed tests. Uncertainty for the goodness-of-fit measures is quantified using nonparametric bootstrap (Efron et al., 1979) from the R package **boot** (Canty and Ripley, 2017). Missing values can be treated using multiple multivariate imputation by chained equations (MICE, Buuren et al., 2006), which is implemented in the R package **mice** (van Buuren and Groothuis-Oudshoorn, 2011). Estimates for the person locations derived from Van Schuur (1988) and Johnson (2006) are available to the user of the package. Generally, the MUDFOLD algorithm is suitable for studies where there are no restrictions on the number of items that a person can "pick". Besides these *pick-any-out-of-N* study designs, sometimes individuals are restricted to select a prespecified number of items, i.e. *pick-K-out-of-N*. The latter design, due to the violation of independence does not respect the IRT assumptions. However, our package is also able to deal with such situations.

Methodology

Consider a sample of n individuals randomly selected from a population of interest in order to take a behavioral test. Participants indexed by i, $i = 1, 2, \ldots, n$ are asked to state if they do agree or do not with each of $j = 1, 2, \ldots, N$ statements (i.e. items) towards a unidimensional attitude θ that we intend to measure. Let X_{ij} be random variables associated with the $0, 1$ response of subject i on item j. We will denote the response of individual i on item j as X_{ij} and x_{ij} its realization.

Subsequently, we can define the IRF for an item j as a function of θ. That is, the probability of positive endorsement of item j from individual i with latent parameter θ_i we write $P_j(\theta_i) = P\left(X_{ij} = 1|\theta_i\right)$. In PIRT models for unfolding, $P_j(\theta_i)$ is a parametric unimodal function of the proximity between the subject parameter θ_i and the item parameter β_j. NIRT unfolding models avoid to impose strict functional assumptions on the IRFs. In the latter case, the focus is on ordering the items on a unidimensional continuum. The item ranks are then used as measurement scale to calculate person specific parameters (ideal-points) on the latent continuum.

Assumptions of the nonparametric unfolding IRT model

In unidimensional IRT models, *unidimensionality* of the latent trait, and *local independence* of the responses are common assumptions. However, the usual assumption of *monotonicity* that we meet in the cumulative IRT models, needs modification in the unfolding IRT where *unimodal* shaped IRFs are considered. For obtaining diagnostic properties for the nonparametric unfolding model, Post and Snijders (1993) proposed two additional assumptions for the IRFs. The assumptions of the nonparametric unfolding model are:

[1]URL: http://CRAN.R-project.org

A1. *Unidimensionality* (UD): There exists a unidimensional latent variable $\theta \in \mathbb{R}$ on which individuals and items are scaled.

A2. *Local Independence* (LI): The responses of individuals on distinct items are independent given the latent parameter θ, i.e the joint conditional probability of N responses simplifies into the likelihood form,

$$P\left(\mathbf{X} = \mathbf{x} \mid \theta = \theta_0\right) = \prod_{j=1}^{N} P_j\left(\theta_0\right)^{x_j} \left[1 - P_j\left(\theta_0\right)\right]^{1-x_j}.$$

A3. *Unimodality* (UM): For every item j, $P_j(\theta)$ is a weakly *unimodal* function of θ.
For the sake of clarity, a function $P_j(\theta) : \mathbb{R} \to \mathbb{R}$, is weakly unimodal if there exists a $\beta_j \in (-\infty, +\infty)$ such that, $P_j(\theta)$ is non decreasing for all $\theta \leq \beta_j$ and non increasing for all $\theta \geq \beta_j$. The location parameter β_j for the jth item is the value of the latent trait for which the IRF $P_j(\theta)$ reaches its maximum (or the midpoint of the interval where $P_j(\theta)$ is maximum when β_j is not unique).

A4. *Stochastic Ordering* (SO): For any probability distribution $G(\theta)$ of latent trait values and any value θ_0 on the latent scale, $P_G\left(\theta > \theta_0 | X_j = 1\right)$ is nondecreasing function of j for all j such that $p_j(x) > 0$.
Given the item ordering this assumption is equivalent to two properties for the IRFs. First, given that a single item is chosen, the posterior densities g of θ have a monotone likelihood ratio (MLR) in θ, and second, the IRFs have a monotone traceline ratio (MTR). The next assumption concerns only unfolding models and is not applicable for cumulative IRT.

A5. *Manifest unimodality* (MUM): For any probability distribution $G(\theta)$ of latent trait values, and for any values $\theta_1 < \theta_2$, the posterior probability $P_G\left(\theta_1 < \theta < \theta_2 \mid X_j = 1\right)$ is a weakly unimodal function of j.

Assumption **A1** implies that there exist only one latent trait that explains the responses of persons on the items. Assumption **A2** is mathematically convenient since it reduces the likelihood to a simple product and implies that given the latent trait value no other information on the other items is relevant to predict the responses to a particular item. The next assumption concerns the conditional distribution of each item given the latent trait. The unimodality assumption that is described in **A3** restricts the IRFs to have a single-peak shape without imposing any explicit functional form. If **A3** holds for all the IRFs then we can order the items on the unidimensional continuum based on their location parameter β_j such that $\beta_1 \leq \beta_2 \leq \cdots \leq \beta_N$. The set of assumptions **A1**-**A3** is the core in unfolding IRT models.

Additionally, two assumptions are needed about the individuals $\{i \mid i = 1, \ldots, n\}$ and the distribution G of their latent trait values $\{\theta_i \mid i = 1, \ldots, n\}$ in order to obtain testable properties for the nonparametric unfolding model (Post and Snijders, 1993). Assumption **A4** is analogous to the invariant item ordering (IIO) assumption in the monotone IRT models and implies that the posterior distribution of θ given a positive response to an item located at β_j is stochastically ordered by the location β_j (Johnson, 2006). In simple words, **A4** assumes that an individual who responds positively to an item with higher rank should have a larger latent trait than those individuals who respond positively to a low-rank item. For example, if a person responds positively to an item that is considered politically conservative, then this person is more likely to be a conservative compared to a person who responded positively to a liberal statement. Despite the fact that this assumption seems intuitive, not all parametric unfolding models require this additional assumption. Assumption **A5** suggests that individual i who endorses item j has a latent trait value θ_i that is most likely close to item location β_j and less likely either much lower or much higher on the latent scale than that. Post (1992) shows that the measurement assumptions **A4**-**A5** are related to the mathematical property of total positivity of order 2 (TP_2) (Karlin, 1968). In addition, if the IRFs $P_j(\theta)$ are positive for all j, then these assumptions hold if and only if the IRFs satisfy the property of TP_3.

Errors and scalability coefficients

PIRT approaches use well defined IRFs that parametrize explicitly persons and items on some known parameter space. Estimates of the parameters can be obtained using suitable frequentist or Bayesian methods and the fit of the model to the data is assessed using goodness-of-fit indices. Contrarily, in NIRT modelling the functional form of the IRF is unknown and alternative estimation methods are needed (Mokken, 1997).

Models in the NIRT class, typically employ item selection algorithms that construct ordinal measurement scales for persons by iteratively maximizing some scalability measure upon the items. The resulting scales are then used to locate the individuals on the latent continuum based on their responses. Usually, these item selection algorithms are bottom-up methods that are divided into two

parts. In the first part the algorithms seek to find the best minimal scale, that is a minimal set of items that meets certain scalability requirements. The best minimal scale is the starting point for the second part of the scaling procedure, where it is extended iteratively by adding in each step the item that best fulfills the prespecified scalability criteria.

As in other NIRT models, MUDFOLD adopts a two step item selection algorithm that identifies the unique rank order for a maximal (sub) set of items. In this algorithm, scalability coefficients analogous to the ones defined by Mokken (1971) are used as tests for the goodness-of-fit. Mokken's coefficients are similar to the H coefficients proposed by Loevinger (1948), which were defined on the basis of violation probabilities of the deterministic cumulative model (see Guttman, 1944) for ordered item pairs. In the same line, the scalability coefficients in MUDFOLD are defined on the basis of violation probabilities of the deterministic unfolding model of Coombs (1964) for triples of items. MUDFOLD's scalability coefficients in a triple of items compare the number of errors observed (i.e. the number of $\{1,0,1\}$ responses, which falsify the Coombsian model) with the number of errors that we would expect if the items were statistically independent. A triple of items is a permutation (ordering) of three distinct items.

Observed errors (O) in an ordered triple of items (h,l,k) with h, l, k distinct elements of the set $\{1,2,\ldots,N\}$, is the frequency of $\{1,0,1\}$ responses over all individuals. The observed errors can be calculated by $\mathrm{O}_{hlk} = \sum_{i=1}^{n} x_{ih}\ (1 - x_{il})\ x_{ik}$ where $x_{i.}$ is the realization of random variable $X_{i.}$ and $x_{i.} = 1$ if the i^{th} individual responds positively on item $(.)$ otherwise $x_{i.} = 0$. It can be seen that the number of observed errors for three items stays invariant for the permutations (h,l,k) and (k,l,h) for any $h \neq l \neq k \neq h$ in the integer set $\{1,2,\ldots,N\}$.

Expected errors (EO) in an ordered item triple (h,l,k) under random ordering is the expected frequency of $\{1,0,1\}$ responses if the items h, l, and k were statistically independent multiplied by the sample size, $\mathrm{EO}_{hlk} = p\,(h)\ (1 - p\,(l))\ p\,(k)\ n$. We can estimate $p\,(j)$ for item j $p\,(j) = \sum_{i=1}^{n} \frac{x_{ij}}{n}$ as the relative frequency for item j.

Scalability coefficient (H) for any ordered item triple (h,l,k), is defined as the value obtained if we subtract from unity the ratio of observed errors over the expected errors for this triple,

$$\mathrm{H}_{hlk} = 1 - \frac{\mathrm{O}_{hlk}}{\mathrm{EO}_{hlk}},\ \forall\, h,l,k \in \{1,2,\ldots,N\}. \tag{1}$$

Using the scalability coefficients for triples, we can extend the notion of scalability for a scale s consisting of m items, where $3 < m \leq N$ and for an item $j \in s$. The H coefficient for an item $j \in s$, $j = 1,2,\ldots,m$ is given by,

$$\mathrm{H}_j\,(s) = 1 - \frac{\sum_{(h,l,k)\in T_j(s)} \mathrm{O}_{hlk}}{\sum_{(h,l,k)\in T_j(s)} \mathrm{EO}_{hlk}}, \tag{2}$$

where $T_j\,(s) = \{(s_h,s_l,s_k) \mid s_h < s_l < s_k : j \in \{s_h,s_l,s_k\}\}$ is the set of all item triples (with respect to the item order), that include item j.

Given that the m items constituting the scale are ordered, we are able to calculate the H coefficient for the total scale s by summing the observed errors and the expected errors for all $\frac{m!}{3!(m-3)!}$ triples of items of s and calculate their error ratio. If we subtract the obtained number from the unity results in a total scalability measure,

$$\mathrm{H}_{\text{total}}\,(s) = 1 - \frac{\sum_{(h,l,k)\in T(s)} \mathrm{O}_{hlk}}{\sum_{(h,l,k)\in T(s)} \mathrm{EO}_{hlk}}, \tag{3}$$

where $T\,(s) = \{(s_h,s_l,s_k) \mid s_h < s_l < s_k\}$ is the set of all item triples for a given scale s.

Perfect fit of the scale to the data yields a scalability coefficient value of $\mathrm{H}_{\text{total}}\,(s) = 1$. The latter means that no error patterns are observed in this scale. Likewise, $\mathrm{H}_{\text{total}}\,(s) = 0$ implies that the number of observed errors is equal to what you would have expected for a random ordering. Values around 0.5 suggest a moderate unfolding scale. Calculating the triple scalability coefficients for all the items is the first step in the construction of a MUDFOLD scale.

We will demonstrate how the H coefficients for triples are calculated using the dataset `ANDRICH` that comes with the **mudfold** package in R data format. The dataset contains the binary responses of $n = 54$ students on $N = 8$ statements towards capital punishment. This attitudinal test have been constructed by Andrich (1988) in order to measure attitudes towards capital punishment.

Calculating scalability coefficients for the `ANDRICH` data. We can install and subsequently load the package and the data into the R environment.

```
## Install and load the mudfold package and the ANDRICH data
install.packages("mudfold")
library(mudfold)
```

```
data("ANDRICH")
N <- ncol(ANDRICH) # number of items
n <- nrow(ANDRICH) # number of persons
item_names <- colnames(ANDRICH) # item names
```

Functions for calculating the observed errors, expected errors, and H coefficients for each possible item triple are available internally in the **mudfold** package. These functions can be accessed by the `:::` operator. For the `ANDRICH` data the H coefficients for triples can be calculated as follows.

```
experr <- mudfold:::Err_exp(ANDRICH) # errors expected
obserr <- mudfold:::Err_obs(ANDRICH)  # errors observed
hcoeft <- 1 - (obserr / experr) # H coefficients
```

Generally, there exist N^3 item permutations of length three with repetitions that can be obtained from N items. Thus, the corresponding H coefficients of each possible item permutation of length three can be stored into a three way array with dimension $N \times N \times N$. In the `ANDRICH` data example, the scalability coefficients for the item permutations of length three are stored into three-way array with dimension $8 \times 8 \times 8$. It can be seen that the H coefficients for symmetric permutations stay invariant and we demonstrate this feature below. Consider the ordered triple of items $(HIDEOUS, DONTBELIEV, DETERRENT)$ and its symmetric permutation $(DETERRENT, DONTBELIEV, HIDEOUS)$.

```
triple_HDODE <- matrix(c("HIDEOUS", "DONTBELIEV", "DETERRENT"), ncol = 3)
triple_DEDOH <- matrix(rev(triple_HDODE), ncol = 3)
```

If we compare the H coefficients of these two (symmetric) triples we will see that they coincide.

```
## Compare H coefficients
hcoeft[triple_HDODE] == hcoeft[triple_DEDOH]
```

The H_{hlk} coefficients form the basis in order to calculate the scalability coefficients for items and scales. The item selection algorithm implemented in the package runs in two steps and scalability criteria are used in both steps.

Scale construction

In the first step of the item selection algorithm, a search in order to find the best triple of items is conducted. A lower bound λ_1 that controls the scalability properties of the best triple can be specified by the user (default value is $\lambda_1 = 0.3$). The value of λ_1 is used as a threshold to determine if the triple is good enough to continue the scaling process. Larger values of λ_1 lead to more strict criteria while lower values of λ_1 relax these criteria.

In its second step, the item selection algorithm extends the best elementary scale repeatedly until no more items fulfill its scalability criteria. A second threshold $\lambda_2 = 0$ is explicitly used in the first criterion of this step. This threshold controls the scalability properties of the triples containing a candidate item in the scale extension procedure. As for λ_1, larger values of λ_2 lead to more strict scalability requirements, while, lower values relax these requirements.

Step 1: search for the best unique triple.

The search for the optimal item triple in the first step requires the calculation of the scalability coefficients for every possible permutation of length 3 that can be obtained from N starting items.

Among the set of all permutations of length three we seek to find those that fulfill certain scalability criteria and we call this set of permutations unique triples. Unique triples is a finite set containing all (h, l, k) with $h, l, k \in \{1, 2, \ldots, N\}$, and $h \neq l \neq k \neq h$ for which only one of their permutations (out of three possible) presents a positive H_{hlk} coefficient i.e.

$$\mathrm{H}_{hlk} > 0, \quad \mathrm{H}_{hkl} < 0, \quad \mathrm{H}_{lhk} < 0.$$

This guarantees that triples in the set of unique triples are "uniquely" represented on the latent dimension, i.e. are scalable together in only one permutation besides the reverse

permutation. From the set of unique triples, the triple (h, l, k) that has the maximum H_{hlk} is called the best unique triple and it will be selected as the best starting scale if its scalability coefficient is positive and greater than a specified lower bound λ_1. If more than one triples fulfill the requirements for being the best unique triple it can be shown that all of them will converge to same solution in the second step.

If the set of unique triples is empty, the algorithm stops automatically without proceeding in the second step. The same holds also in the case in which unique triples exist but their scalability coefficient is lower that the bound specified by the user.

First step: search for best minimal scale in the `ANDRICH` data. Here we describe how the main function of the **mudfold** package searches for the best minimal unfolding scale in the first step of the implemented algorithm. After we calculated the observed errors, the expected errors, and the scalability coefficients for each triple of items in the `ANDRICH` dataset, we need to determine the optimal triple for the first step of MUDFOLD's item selection algorithm. The triples of items in the order (h, l, k) for the `ANDRICH` data can be obtained with the `combinations()` function from the R package **gtools** (Warnes et al., 2015). These combinations are then permuted twice to yield the orderings (h, k, l) and (l, h, k) respectively.

```
## Install and load the library "gtools"
install.packages(gtools)
library(gtools)

## Obtain item permutations (h,l,k), (h,k,l), and (l,h,k)
perm1 <- combinations(N, 3, item_names, set = FALSE)
perm2 <- perm1[, c(1,3,2)]
perm3 <- perm1[, c(2,1,3)]
```

The set of unique triples can then be obtained.

```
## Find the set of unique triples.
unq <- rbind(perm1[(hcoeft[perm1] > 0 & hcoeft[perm2] < 0 & hcoeft[perm3] < 0), ],
             perm2[(hcoeft[perm1] < 0 & hcoeft[perm2] > 0 & hcoeft[perm3] < 0), ],
             perm3[(hcoeft[perm1] < 0 & hcoeft[perm2] < 0 & hcoeft[perm3] > 0), ])
```

The set of unique triples in the `ANDRICH` data example contains sixteen item triples. With the command `hcoeft[unq]` we can see that all except one of the triples show H_{hlk} coefficients greater than the lower bound. The ordered triple of items $(INEFFECTIV, DONTBELIEV, DETERRENT)$ is selected as the best starting scale with a maximum scalability coefficient of 0.853 which is indeed larger than λ_1. This triple will be extended repeatedly in the second step of the algorithm. In each iteration one from the remaining ones is added to the scale in a specific position if certain scalability requirements are met.

Step 2: extending the best starting scale

Given the best unique triple obtained in the first step of the algorithm, in the second step of the item selection process the algorithm investigates repeatedly the remaining $N - 3$ items to find the best fourth, fifth, etc to add to the scale. In each iteration of this step, all the possible scales that contain one of the remaining items in every possible position are investigated to choose the most appropriate one.

For a scale consisting of m items, $(3 \leq m \leq N - 1)$ we intend to find one of the remaining $N - m$ items to add in the scale. For the $(m + 1)^{th}$ item there exist $m + 1$ possible scale positions that have to be investigated with respect to their scalability properties. In each iteration of the MUDFOLD scaling algorithm, the number of candidate scales under investigation is $(N - m)(m + 1)$.

In order to determine the $(m + 1)^{th}$ best fitting item we test three criteria. The first criterion uses an explicit value λ_2 (by default $\lambda_2 = 0$) as a lower bound for the scalability coefficients. The scalability criteria in the second step are :

1. All the $\binom{m}{2}$ item triples in the scale (with respect to the item order), containing the candidate item must have H_{hlk} coefficient greater than λ_2.

2. If more than one item fulfills the first criterion, then the item with the minimum number of possible scale positions is chosen.

3. The scalability coefficient $H_j\,(s)$ of the selected item has to be higher than λ_1.

It can be the case that more than one scales fulfill these criteria. In such instances, the algorithm continues by choosing the scale that includes the most uniquely represented item and shows the minimum number of expected errors. The scale extension process continues as long as the scalability criteria described above are fulfilled.

Second step: scale extension for the `ANDRICH` data For the `ANDRICH` data, after the first step of the item selection process where we obtained the best unique triple, the remaining five items can still be added to the scale.

```
BestUnique <- unq[which.max(hcoeft[unq]), ] # Best unique triple
ALLitems <- colnames(ANDRICH)
Remaining <- ALLitems[!ALLitems %in% BestUnique] # Remaining items
```

Next, an iterative procedure needs to be defined for the second, scale extension step of the MUDFOLD algorithm. Adding one item in each repetition implies that a maximum of $N - 3 = 5$ iterations can take place if all items fit in a MUDFOLD scale. In each iteration we construct the scales to be evaluated where each scale contains one of the remaining items in a specific position.

For example, in the first iteration of the scale extension step for the `ANDRICH` dataset, all the scales that need to be assessed can be constructed as follows. First we need to consider all the possible positions where a new item can be added. The possible positions depend on the length of the existing scale. At this point, since the scale consists of three items there exist four possible positions where a new item can be added.

```
## Create indices to be used in constructing scales
lb <- length(BestUnique) # length of best unique triple
lr <- length(Remaining) # number of remaining items to add in the scale

## create all possible positions where each new item from Remaining
## can be added in the scale
index_rep <- rep(seq(1, (lb+1)) ,lr) - 1 # possible positions
index_irep <- rep(Remaining, each = lb+1) # item for each position
```

After we define all the possible positions for new items, each item is added in every position and results in a different scale to be assessed.

```
## Create all possible scales by adding each item in Remaining
## to every possible position of BestUnique
ALLscales <- lapply(1:length(index_rep),
         function(i) append(BestUnique, index_irep[i], after = index_rep[i] ))
```

Each of these scales will be judged in terms of its scalability properties. For instance, let us consider the first scale that is constructed in the first iteration of the scale extension step in the `ANDRICH` data.

```
Examplescale <- ALLscales[[1]]
Examplescale
# "HIDEOUS"    "INEFFECTIV" "DONTBELIEV" "DETERRENT"
```

This scale has been constructed after inserting the item *HIDEOUS* into the first possible position of the minimal scale *(INEFFECTIV, DONTBELIEV, DETERRENT)*. The first scalability criterion for this scale determines if the H_{hlk} coefficients of the triples that contain the new item (i.e. *HIDEOUS*) are larger than a user specified λ_2 (default $\lambda_2 = 0$). We can extract all the triples for this specific scale using the `combinations()` function.

```
les <- length(Examplescale)
ExamplescaleTRIPLES <- combinations(n = les, r = 3, v = Examplescale, set = FALSE)
```

From the four triples in total, only the first three are containing the new item *HIDEOUS*. We can obtain the H coefficient for each of these triples with

```
hcoeft[ExamplescaleTRIPLES[1:3, ]]
```

and we can see that the triple (*HIDEOUS, INEFFECTIV, DETERRENT*) has a H coefficient which is lower than λ_2. Hence, this scale does not fulfill the first criterion and should be excluded from the scale extension process. The first criterion is calculated for every scale possible and the scales that conform to this criterion continue the scale extension process. Lowering the values of λ_2 to a negative number will allow more scales to pass this criterion, while setting λ_2 to a large negative number e.g. -99 will allow all scales to pass this criterion.

The second scale assessment determines which scale or scales contain the item that is the most "uniquely" represented. Let us assume that the number of scales that fulfill the first criterion is six. Moreover, assume that five out of these six scales contain the item *MUSTHAVEIT* and one scale contains the item *CRIMDESERV*. In this scenario the scale that contains the item *CRIMDESERV*, will be the one that continues the scale extension.

The scales that contain the least frequently observed item are checked according to a third criterion. The third and last criterion in the iterative scale extension phase concerns the scalability properties of the new item. The scale that contains the new item with the highest item scalability coefficient will be chosen as the best MUDFOLD scale if and only if $\mathrm{H}_j\,(s) > \lambda_1$ where λ_1 is the lower bound that have been used also in the first step of the item selection algorithm.

In the `ANDRICH` example the algorithm completes five iterations in the second step which means that all the items are included in the MUDFOLD scale. The latter, consists of eight items and shows a scale scalability coefficient equal to 0.64.

After a MUDFOLD scale with a good fit is obtained, one can assess its unfolding quality. This is done by scale diagnostics described by Post (1992) and Post and Snijders (1993). These diagnostics are based on sample proportions from which the unimodality assumption of the scale is evaluated and nonparametric estimates of the item response functions are obtained.

MUDFOLD diagnostics

In this section, we discuss diagnostics implemented in the **mudfold** package, which can be used to assess if a scale s consisting of m items, $j = 1, \ldots, m$ conforms with the assumptions **A2** to **A5** of a unidimensional nonmonotone homogeneous MUDFOLD scale.

Diagnostic for assumption A2

Let us denote by $\mathbf{X}_{-j}$ the $n \times (m-1)$ matrix that contains the responses of n individuals to all the items in the scale except item j. Testing if **A2** (local independence) holds, is equivalent to testing if the positive response on an item depends solely on the latent trait θ, i.e. $P\left(X_j = 1|\mathbf{X}_{-j}, \theta\right) = P\left(X_j = 1|\theta\right)$. If $p_j = P\left(X_j = 1\right)$ denotes the probability of positive response to item j, testing this hypothesis implies fitting the following regularized logistic regression model,

$$\log \frac{p_j}{1-p_j} = \beta_0 + \sum_{k=1}^{m-1} \beta_k X_{-jk} + \beta_\theta \hat{\theta}, \tag{4}$$

where X_{-jk} denotes the kth column of $\mathbf{X}_{-j}$ and $\hat{\theta} = \left(\hat{\theta_1}, \ldots, \hat{\theta_n}\right)$ is a nonparametric estimate of the latent attitude with regression parameter β_θ. The response regression parameters β_k are penalized using the least absolute shrinkage and selection operator (LASSO, Tibshirani, 1996). LASSO shrinks the coefficients β_k of the regression in (4) towards zero. If $\beta_k = 0$ for all $k = 1, \ldots, m$ then the local independence assumption if fulfilled and the probability of positive response on the item j depends only on θ. On the other hand if there is any k for which $\beta_k \neq 0$ there is evidence of violations in the local independence assumption. Fitting sparse generalized linear models with simultaneous estimation of the regularization parameter is straightforward in R with the function `cv.glmnet()` that is available with the package **glmnet** (Friedman et al., 2010).

Diagnostic for assumption A3

The condition **A3** required by MUDFOLD is the assumption of unimodality of the IRFs, which are unknown nonlinear functions of the latent trait. In order to obtain estimates of these functions, we use a nonlinear generalized additive model (GAM, Wood, 2011) that is implememented in the R package **mgcv** (Wood, 2017). Specifically, for each item the probability of positive response p_j is modelled as a smooth function of the latent trait θ, that is,

$$\log \frac{p_j}{1-p_j} = \beta_0 + \beta_\theta f_\theta\left(\hat{\theta}\right), \tag{5}$$

where $f_\theta\left(\hat{\theta}\right)$ is a smooth function of $\hat{\theta}$. Plotting the probability of positive response modelled by (5) against a nonparametric estimate of the latent trait $\hat{\theta}$, should yield a single 'peaked' curve if the unimodality assumption for the IRFs holds.

Diagnostics for assumptions A4 and A5

For the assumptions **A4-A5**, diagnostic statistics that quantify to which extent the scale agrees with these assumptions have been proposed by Post (1992). These statistics are based on conditional IRF probabilities, which are estimated by their corresponding sample proportions and collected into a matrix that is called the conditional adjacency matrix (CAM).

CAM in its (j,k) element contains the conditional frequency that a subject from the sample will choose the row item j given that the column item k is chosen. The probability $P\left(X_j = 1 \mid X_k = 1\right)$ is estimated from the data by dividing the joint frequency of choosing both items j and k by the relative frequency of choosing item k. That is,

$$\text{CAM}_{jk} = \frac{\sum_{i=1}^{n} x_{ij}\, x_{ik}\ /n}{\sum_{i=1}^{n} x_{ik}\ /n} = \frac{\sum_{i=1}^{n} x_{ij}\, x_{ik}}{\sum_{i=1}^{n} x_{ik}}, \text{ for } j \neq k. \tag{6}$$

In the package **mudfold**, the CAM can be obtained using the function `CAM()`, which takes as input either a fitted MUDFOLD object or a dataset with the complete responses of n individuals to m items. In the `ANDRICH` dataset example, the CAM of the original data can be calculated using the command `CAM(ANDRICH)`.

Each row of the CAM is regarded as an empirical estimate of the corresponding IRF. Hence, if the ordering of the items is correct, and if assumptions **A1** to **A5** hold, then (i) the observed maxima of the different rows of the CAM are expected to appear around the principal diagonal (moving maxima property), and (ii) the rows of the CAM are expected to show a weakly unimodal pattern. One can potentially evaluate the unfolding model by checking how strongly the observed row patterns of the CAM deviate from the expected patterns described above.

Max statistic (MAX) : The moving maxima property of the CAM corresponds to condition **A4**, which assumes stochastic ordering of the items by their location parameter β_j. In order to formally check this assumption, Post (1992) proposed a statistic that quantifies the violations of the moving maxima property for the rows of the CAM , which is called the max statistic (MAX).

Calculation of the MAX can be done in two ways, namely a top-down and a bottom-up method

$$\text{MAX}_j = \begin{cases} \sum_{k=j+1}^{m} \max\left(0, \left(M_j - M_k\right)\right) & \text{(top-down method)} \\ \\ \sum_{k=1}^{j-1} \max\left(0, \left(M_k - M_j\right)\right) & \text{(bottom-up),} \end{cases} \tag{7}$$

where M_j is the position of the maximum in the jth row of CAM. In order to create a measure of the moving maxima property that is bounded within the interval $[0,1]$ we divide MAX_j by the number of potential violations of the moving maxima property which are approximately equal to $m^2/12$.

The sum over all rows yields the total MAX statistic of the scale, i.e. $\text{MAX}_{\text{total}} =$

$\sum_{j=1}^{m} \text{MAX}_j$.. The quantity $\text{MAX}_{\text{total}}$ will be the same for both methods in (7), however, the number of items showing positive MAX can be different. In this situation the method that yields the minimum number of items showing positive MAX is chosen. If the number of items with positive MAX is the same for both methods then we choose arbitrarily the top-down method. In the case where M_j is next to a diagonal element then the maximum in the jth row can have two positions and the position that yields the lower MAX value will be chosen.

The MAX statistic can be calculated using the function `MAX()` from the R package **mudfold**, which takes as input either a fitted MUDFOLD object obtained from the main `mudfold()` function, or an object of class `"cam.mdf"` calculated from the function `CAM()`. The argument `'type'` of the `MAX()` function controls if the MAX for the items or the whole scale will be returned to the user. Visual inspection of the observed maxima pattern can also be useful. If the maximum values of the CAM rows are close to the diagonal then the unfolding model holds. The `diagnostics()` will return and plot a matrix with a star at the maximum of each CAM row for visual inspection of their distribution.

Iso statistic (ISO) : In order to quantify if the rows of the CAM show a weakly unimodal pattern, the iso statistic (proposed by I. Molenaar, personal communication) was introduced. Iso statistic (ISO), is a measure for the degree of unimodality violation in the rows of CAM. ISO can be obtained for each item (ISO_j) and their summation results in the total ISO for the scale (ISO_{tot}).

To come up with an ISO value for an item j, one should first locate the maximum in each row of the CAM. If we index m^* the maximum in row j of CAM, the ISO measures deviations from unimodality to the left and right of m^*, i.e.

$$\text{ISO}_j = \sum_{h \le k \le m^*} \max\left(0, \text{CAM}_{jh} - \text{CAM}_{jk}\right) + \sum_{m^* \le h \le k} \max\left(0, \text{CAM}_{jk} - \text{CAM}_{jh}\right). \quad (8)$$

The total ISO statistic for a scale consisting of m items is calculated as the sum of the individual ISO statistics, i.e. ISO_j's, i.e. $\text{ISO}_{\text{total}} = \sum_{j=1}^{m} \text{ISO}_j$. The ISO statistic, both for an item or for the scale, is zero if the unimodality in row j of the conditional adjacency matrix is not disturbed and positive if disturbances in unimodality occur in row j.

The user can calculate the ISO statistic using the function `ISO()`, which takes as input outputs either from the `mudfold()` function, or from the function `CAM()` and returns a vector with the ISO_j's for each $j \in \{1, 2, \ldots, m\}$ or the sum of this vector if `type = 'scale'`.

All the diagnostic tests discussed in this section are implemented in the function `diagnostics()` of the **mudfold** package. The function `diagnostics()` can be used with fitted objects from the main `mudfold()` function.

Uncertainty estimates for MUDFOLD statistics

Since the sampling distributions of the MUDFOLD's goodness-of-fit and diagnostic statistics are non-standard, calculating their standard errors is not straightforward. Instead, for providing uncertainty estimates of the MUDFOLD statistics both at the item and the scale level, nonparametric bootstrap is used (Efron et al., 1979). Bootstrap is a resampling technique that can be used for assessing uncertainty in instances when statistical inference is based on complex procedures. With bootstrapping we sample R times n samples with replacement from a dataset of size n. The bootstrap samples of the statistic obtained from R iterations are then used to approximate the sampling distribution of the statistic.

Given a MUDFOLD scale s, statistics for items such as the $\text{O}_j(s)$, $\text{EO}_j(s)$, $\text{H}_j(s)$, and the total scale such as the O_{total}, EO_{total}, H_{total} are bootstraped R times. The bootstrap procedure implemented in **mudfold** depends on the function `boot()` from the R package **boot** (Canty and Ripley, 2017). Using the **boot** package allows the user of **mudfold** package to obtain different types of confidence intervals for assessing uncertainty using the function `boot.ci()`.

Additional to the uncertainty estimates, a bootstrap estimate of the unfolding scale can be also calculated. This estimate corresponds to the most frequently obtained MUDFOLD

scale in R bootstrap iterations. In many instances the bootstrap estimate will coincide with MUDFOLD scale obtained by the item selection algorithm. When the two estimates are different the bootstrap scale estimate can be used to correct the MUDFOLD scale after assessing its properties carefully.

Nonparametric estimation of person ideal points

With MUDFOLD, after obtaining an item ordering (scale) that consists of a (sub) set of m items, $m \leq N$, one can estimate in a nonparametric way subject locations on a latent continuum. Two nonparametric estimators can be used with slightly different properties both based on the Thurstone (1927, 1928) estimator for the measurement of attitudes.

Originally, the Thurstone estimator $\hat{\theta}_i^{\beta}$ of the i-th respondent location parameter given a vector of known item location parameters $\beta = (\beta_1, \beta_2, \ldots, \beta_m)^{\mathsf{T}}$ was defined as,

$$\hat{\theta}_i^{\beta} = \frac{\sum_{j=1}^{m} \beta_j x_{ij}}{\sum_{j=1}^{m} x_{ij}}, \tag{9}$$

where x_{ij} is the response of person i on item j. The parameter estimate $\hat{\theta}_i^{\beta}$ for each i takes values within the item parameter range. In MUDFOLD however, the item parameters vector β is unknown, thus we need to estimate it. In order to do so, we make use of two alternative estimates for β's proposed by Van Schuur (1988) and Johnson (2006), respectively. The former uses item ranks as approximations of the item locations while latter uses item quantiles.

Van Schuur's person parameter estimator uses the item ranks obtained from MUDFOLD's item selection algorithm as estimates for the vector $\beta = (\beta_1, \beta_2, \ldots, \beta_m)^{\mathsf{T}}$. Since MUDFOLD estimates only the rank order of the parameter vector, i.e. $r = (r_1, r_2, \ldots, r_m)^{\mathsf{T}}$ one can define a rank estimate

$$\hat{\beta}_j^{r} = r_j, \tag{10}$$

where r_j is the rank of the item j on the MUDFOLD scale. By using the estimated ranks as approximations of the parameter vector we can estimate a respondent's location as the mean of the endorsed item ranks. That is,

$$\hat{\theta}_i^{r} = \begin{cases} \frac{\sum_{j=1}^{m} r_j x_{ij}}{\sum_{j=1}^{m} x_{ij}}, & \text{if } \sum_{j=1}^{m} x_{ij} > 0 \\ \\ \text{undefined}, & \text{if } \sum_{j=1}^{m} x_{ij} = 0. \end{cases} \tag{11}$$

Alternatively Johnson's quantile estimator bounds both estimates for θ's and β's within a unit interval. This estimator uses the item ranks divided by the length of the scale m as approximations for the β vector. In all the estimators described in this section, no estimates can be defined for individuals with total score $X_i^{+} = \sum_{j=1}^{m} x_{ij}$ equal to zero. These individuals are not endorsing any item and therefore provide no information whether they belong to the extreme right of the scale or to extreme left. The user of the package **mudfold** can choose between Van Schuur's and Johnson's estimators for obtaining persons scores on the factors.

Missing values

Missing data occur when intended responses from one or multiple persons are not provided. Handling missing values is critical since it can bias inferences or lead to wrong conclusions. One way to go is to ignore the missing observations by applying list-wise deletion. This, however, can lead to a great loss of information especially if the number of missing values is large. The other approach, is to replace the missing values with actual values which is called imputation.

In the case of random missing value mechanisms such as missing completely at random (MCAR) and missing at random (MAR) (Rubin, 1976; Little and Rubin, 1987), different approaches can be used in order to impute the missing observations. Imputation within IRT

is in general associated with more accurate estimates of item location and discrimination parameters under several missing data generating mechanisms (Sulis and Porcu, 2017). In the package **mudfold** missing values can be imputed using the logistic regression version of multiple multivariate imputation by chained equations (MICE). The latter is available from the R package **mice**. MICE imputation within **mudfold** can be used solely or in combination with bootstrap uncertainty estimates. In the latter case, each bootstrap sample is imputed before fitting a MUDFOLD scale, while in the former the data are imputed M times and the results are averaged across the M datasets.

The mudfold package

The R package **mudfold** contains a collection of functions related to the MUDFOLD item selection algorithm. In the following we describe the functionality of the package and the `ANDRICH` dataset is used for demonstration purposes.

Description of the functions `mudfold()` and `as.mudfold()`

The main function of this package, called `mudfold()`, fits Van Schuur's item selection algorithm to binary data in order to obtain a unidimensional ordinal scale for the persons. The `mudfold()` function can be called with,

```
mudfold(data, estimation, lambda1, lambda2, start.scale,
nboot, missings, nmice, seed, mincor, ...)
```

The functions has ten main arguments where only the first one is obligatory. These are:

`data`: The input data, i.e. a $n \times N$ `data.frame` or `matrix`, with persons in the rows and items in the columns. It contains the binary responses of n individuals on N items. .

`estimation`: This argument handles the nonparametric estimation of the person parameters. The default, `estimation = "rank"` uses a rank based estimator (Van Schuur, 1988). Alternatively, person parameters are obtained by a quantile estimator (Johnson, 2006), which is accessible by setting `estimation = "quantile"`.

`lambda1`: The parameter λ_1, $0 \leq \lambda_1 \leq 1$ is a user specified lower bound for scalability criteria that are used in MUDFOLD's item selection algorithm. In the default setting, $\lambda_1 = 0.3$. Large values of λ_1 lead to more strict criteria in the item selection procedure.

`lambda2`: Parameter λ_2, $-\infty < \lambda_2 \leq 1$ is a lower bound explicitly used at the first scalability criterion of the second step (default $\lambda_2 = 0$).

`start.scale`: The user can pass to this argument a character vector of length greater than or equal to three, containing ordered item names from `colnames(data)` that are used as the best elementary scale for the second step of the item selection algorithm. If `start.scale = NULL` (default), the first step of the item selection algorithm determines the best elementary triple of items that is extended in the second step.

`nboot`: Argument that controls the number of bootstrap iterations. If `nboot = NULL` (default) no bootstrap is applied.

`missings`: Argument that controls treatment of missing values. If `missings = "omit"` (default) list-wise deletion is applied to `data`. If `missings = "impute"` then the mice function is applied to `data` in order to impute the missings `nmice` times.

`nmice`: Argument that controls the number of mice imputations (This argument is used only when `missings = "impute"` and `nboot = NULL`.

`seed`: Argument that is used for reproducibility of bootstrap results.

`mincor`: This can be scalar, numeric vector (of size `ncol(data)`) or numeric matrix (square, of size `ncol(data)` specifying the minimum threshold(s) against which the absolute correlation in the data is compared. See `?mice:::quickpred` for more details. To be used when mice becomes problematic due to co-linear terms.

. . . : Additional arguments to be passed into the `boot()` function (see `?boot` in R).

The function `mudfold()` internally has four main steps. A data checking step, the first step of the item selection process, the second step of the item selection process, and the bootstrap step if the user chooses this option. The output of `mudfold()`, is a `list()` of class `"mdf"` that contains information for each internal step of the function. The first element of the output list contains information on the function call. The second element contains results of the data checking step. The next element of the output contains descriptive statistics obtained from the observed data and the last element of the output has all the information from the the fitting process (triple statistics, first step, second step). If bootstrap is applied to estimate uncertainty , an additional element that contains the bootstrap information is given to the output.

For example, if you want to fit a MUDFOLD scale to the `ANDRICH` data and run a non-parametric bootstrap with $R = 100$ iterations in parallel, you can specify it directly into the `mudfold()` function as follows.

```
fitANDRICH <- mudfold(ANDRICH, nboot = 100, parallel = "multicore", seed = 1)
```

In the example above, the first two arguments are core in the `mudfold()` function. The third argument `parallel` is an argument of the `boot()` function that runs bootstrapping in parallel fashion in order to reduce computational time. The last argument `seed` is used to ensure reproducibility of the bootstrap results.

In some cases the unfolding scale could be known. In these instances, the user is interested in obtaining the MUDFOLD goodness-of-fit and diagnostic statistics for the given scale. The function `as.mudfold()` can be used for treating the given rank order of the items as a MUDFOLD scale. The function uses only the first two arguments of the `mudfold()` function. In principle, this function transforms a given scale into an S3 class `"mdf"` object.

Description of the generic functions

For `"mdf"` objects from the `mudfold()` or `as.mudfold()` functions, generic functions for `print()`, `summary()` and `plot()` and `coef()` are available. The generic function `print.mdf()` can be accessed with,

```
print(x)
```

where `x` is an `"mdf"` class object. This function prints information for `x`, such as time elapsed for fitting, warnings from the data checking step, convergence for each step of the algorithm and statistics with bootstrap confidence intervals if `nboot` is not equal to `NULL`.

In the `ANDRICH` data example, the command `print(fitANDRICH)` is used to print information from the `fitANDRICH` object to the console. The function call together with the elapsed time to fit the model, the number of individuals, and the number of items used in the analysis is the first part of the output. Next, the values of the `mudfold()` arguments are given, which are followed by convergence indicators for each step of the item selection algorithm. Scale statistics such as the scalability coefficient and the ISO statistic are also printed together with their percentile confidence intervals obtained in 1000 bootstrap iterations. The summary of the bootstrap iterations finalize the output when printing the `fitANDRICH` object.

The function `summary` is a generic function that is summarizing information from model fitting functions. In our case the output of `summary.mdf()` is a list object summarizing results from the `mudfold()` function. The function can be called via

```
summary(object, boot, type = "perc", ...)
```

and consists of three arguments:

`object`: a list of class `"mdf"`, output of the `mudfold()` function.

`boot`: logical argument that controls if bootstrap confidence intervals and bootstrap summary for each coefficient will be returned. If `boot = FALSE` (default) no information for bootstrap is returned. When `boot=TRUE`, confidence intervals, standard errors, biases, calculated from the bootstrap iterations for each parameter are given with the output.

type: The type of bootstrap confidence intervals to be calculated if the argumnet boot = TRUE. Available options are "norm", "basic", "perc" (deafult), and "bca". See the argument type of the boot.CI() for details.

The output of the summary.mdf() is a list with two main components. The first component of the list is a data.frame with scale statistics and the second component is a list with item statistics.

Typing summary(fitANDRICH,boot = TRUE) into the R console will return the summary of the fitted scale to the ANDRICH data. The output consists of six distinct data.frame objects. The first data.frame contains information on scale statistics with their bootstraped statistics. The next four data.frame objects correspond to the H coefficients, the ISO statistics, the observed errors, and the expected errors for each item in the scale together with their bootstrap summary statistics. The last data.frame gives descriptive statistics for the items in the scales.

A generic function for plotting S3 class "mdf" objects is also available to the user. The function plot.mdf() returns empirical estimates of the IRFs, the order of the items on the latent continuum or a histogram of the person parameters . You can plot "mdf" class objects with the following R syntax.

```
plot(x, select = NULL, plot.type = "IRF")
```

This function consists of three arguments from which the first is the usual argument x which stands for the "mdf" object to be plotted. The argument plot.type controls the type of plot that is returned, and three types of plots are available. If plot.type = "scale", a unidimensional continuum with the items in the obtained rank order is returned. In the default settings of this function (i.e. plot.type = "IRF"), the corresponding plot has the items on the x-axis indicating their order on the latent continuum and the probability of a positive response on the y-axis. The IRF of each item among the latent scale is plotted with different colours. When plot.type = "IRF" will return a plot with the distribution of person parameters on the latent continuum. The argument select is optional and provides the possibility for the user to plot a subset of items. The user can provide in this argument a vector of item names to be plotted. If select = NULL, the function returns the estimated IRFs for all items in the obtained MUDFOLD scale. For plotting S3 class "mdf" objects, we use the functions na.approx(), melt() and ggplot() from the R packages **zoo** (Zeileis and Grothendieck, 2005), **reshape2** (Wickham, 2007), and **ggplot2** (Wickham, 2009), respectively.

A generic coef.mdf() function for S3 class "mdf" objects can also be used. This function is a simple wrapper that uses a single argument named 'type'. The coef.mdf() will extract nonparametric estimates of: persons ranks when type = "persons", item ranks when type = "items", or both when type = "all" from a fitted MUDFOLD object.

The diagnostics() function

After a scale has been obtained, scale diagnostics need to be applied is order to assess its unfolding properties. The MUDFOLD diagnostics described in section 2.2.4 of this paper are implemented into a function named diagnostics() that can calculate all of them simultaneously. The function syntax is,

```
diagnostics(x, boot, nlambda, lambda.crit, type, k, which, plot)
```

and uses eight arguments described below.

x: a list of class "mdf", output of the mudfold() function.

boot: logical argument that controls if bootstrap confidence intervals and summary for the H coefficients and the ISO and MAX statistics will be returned. If boot = FALSE (default) no information for bootstrap is returned. When boot = TRUE, confidence intervals, standard errors, biases, calculated from the bootstrap iterations for each diagnostic are given with the output.

nlambda: The number of regularization parameters to be used in cv.glmnet() function when testing local independence.

`lambda.crit`: String that specifies the criterion to be used by cross-validation for choosing the optimal regularization parameter. Available options are "class" (default), "deviance", "auc", "mse", "mae". See the argument `'type.measure'` in the `cv.glmnet()` function for more details.

`type`: The type of bootstrap confidence intervals to be calculated if the argumnet `boot = TRUE`. Available options are `"norm"`, `"basic"`, `"perc"` (deafult), and `"bca"`. See the argument type of the `boot.CI()` for details.

`k`: The dimension of the basis in the thin plate spline that is used when testing for IRF unimodality. The default value is `k = 4`.

`which`: Which diagnostic should be returned by the function. Available options are `"H"`, `"LI"`, `"UM"`, `"ISO"`, `"MAX"`, `"STAR"`, `"all"` (default).

`plot`: Logical. Should plots be returned for the diagnostics that can be plotted? Default value is `plot = TRUE`.

For the `ANDRICH` data example the command `diagnostics(fitANDRICH)` will calculate and plot the scale diagnostics for the `fitANDRICH` object.

Unfolding data simulation and description of the `mudfoldsim()` function

In order to provide the user the flexibility of simulating unfolding data, the function `mudfoldsim()` is available from the **mudfold** package. The responses of subjects on distinct items are simulated with the use of a flexible parametric IRF that generalizes proximity relations between item and person parameters.

Assume that we want to simulate a test dataset with responses from n individuals indexed by $i = 1, 2, \ldots, n$ on N proximity items (indexed by j) with latent parameters θ_i and β_j respectively. The vector of item parameters $\beta = (\beta_1, \ldots, \beta_N)^\intercal$ is drawn at random from a standard normal distribution. For the person parameters, the user can choose if they will follow a standard normal distribution, or they will be drawn uniformly in the range of item parameters. Simulating person parameters from a standard normal distribution may imply that a number of individuals are located too far to the left or right of the most extreme items (due to sampling variation). These subjects will not agree with any item. These responses are not useful in unfolding analysis since no discriminant information is provided for the items in the scale. The user of **mudfold** package is free to include or exclude such type of responses.

Unfolding models are also known as distance models since they model the probability of positive endorsement of item j from individual i as a function of the proximity between θ_i and β_j. We consider a linear transformation τ_{ij} of the squared difference $d_{ij}^2 = \left(\theta_i - \beta_j\right)^2$ given by $\tau_{ij} = \gamma_1 + \gamma_2 d_{ij}^2$, where the parameters γ_1 (deterministic parameter) and γ_2 (discrimination parameter) are fixed.

Using τ_{ij} with the standard logistic function one obtains a parametric IRF $f\left(\tau_{ij}\right) = \frac{1}{1+e^{-\tau_{ij}}}$. Consequently, the positive binary response of individual i on item j can be considered as the outcome of a Bernoulli trial with "success" probability $1/\left(1 + e^{-\tau_{ij}}\right)$. Hence, the item response variables X_{ij} that contain binary responses from n individuals on N items, follow a Bernoulli distribution according to,

$$X_{ij} \sim \text{Bernoulli}\left(\frac{1}{1+e^{-\tau_{ij}}}\right) \text{ for } i = 1, \ldots, n,\ j = 1, \ldots, N. \tag{12}$$

In `mudfoldsim()` function, the model parameters $\gamma\,(.)$ are user specified with default settings $\gamma_1 = 5$ and $\gamma_2 = -10$ respectively. This specific set up of the model parameters produces nearly deterministic response curves for the subjects which in turn guarantees that the number of observed errors is small.

We note that the IRF proposed by Andrich (1988) is a special case of the one implemented in the `mudfoldsim()` function for $\gamma_1 = 0$ and $\gamma_2 = -1$. This parametric simulation method is implemented in a flexible R function available from the **mudfold** package. This function

consists of several arguments that allow the user to control the unfolding properties of the simulated data. The function in its default settings can be called easily with the following syntax,

```
mudfoldsim(N, n, gamma1 = 5, gamma2 = -10, zeros = FALSE, parameters = "normal",
seed = NULL)
```

and makes use of six user-specified arguments:

`N`: An integer corresponding to the number of items to be simulated.

`n`: The number of persons to be simulated.

`gamma1`: This argument is passed to the IRF. Controls the γ_1 or discriminative parameter of the IRF. The higher the parameter the larger the number of items that individuals tend to endorse if parameter γ_2 is kept constant.

`gamma2`: The deterministic parameter (i.e. γ_2) of the IRF. As the value of this parameter decreases, individuals tend to make less "errors" in their responses (i.e. their responses are more in line with the unfolding scale).

`zeros`: A logical argument that controls if individuals who endorse no items will be simulated. If `zeros=TRUE` the function allows for individuals that are not endorsing any of the items. On the other hand, if `zeros=FALSE` (default) only individuals who endorse at least one item will be part of the simulated data.

`parameters`: Argument for the person parameters with two options available. In the default option `parameters="normal"` and in this case the person parameters are drawn from a standard normal distribution. On the other hand, the user can set this argument equal to `"uniform"` which implies that subject parameters will be drawn uniformly in the range of the item parameters.

`seed`: An integer to be used in the `set.seed()` function. If `seed=NULL` (default), then the seed is not set.

The output of the `mudfoldsim()` function is a list containing the simulated data (in a random item order), the parameters used in the IRF, and the matrix of probabilities under which the binary data has been sampled.

Description of the `pick()` function

Since the main `mudfold()` function is designed for dichotomous (binary) items, we provide the user with the function `pick()`. The latter, is used to transform quantitative or ordinal type of variables into a binary form. The underlying idea of this function is that the individual selects those items with the highest preference. This transformation can be done in two different ways, either by user specified cut-off value(s) or by assuming a pick K out of N (individuals are asked to explicitly pick K out of N items) response process, where each response vector consists of the K highest valued items. Dichotomization is performed row-wise by default, however the user can also perform the transformation column-wise.

The R function `pick()` can be utilized with the following code,

```
pick(x, k = NULL, cutoff = NULL, byItem = FALSE)
```

and makes use of four parameters. These are,

`x`: A `data.frame` or `matrix` with persons in the rows and items in the columns containing quantitative or ordinal type of responses from n individuals/raters on N items. Missing values are not allowed.

`k`: This integer ($1 \leq$ `k` $\leq N$) controls the number of items a person can pick (default `k=NULL`). This argument is used if one wants to transform the data into pick K out of N form. If the parameter `k` is provided by the user, then `cutoff` should be `NULL` and vice verca.

`cutoff`: The numeric value(s) that will be used as thresholds for the transformation (default `cutoff=NULL`). Any value greater than or equal to the `cutoff` will be 1 and 0 otherwise. The length of this argument should be equal to 1 (indicating same threshold for all rows of `x`) or equal to n (when `byItem=FALSE`) which imposes an explicit cut-off value for each individual in `x`. If `byItem=TRUE` then the length of this parameter should be 1 (global cut-off value) or N (explicit cut-off per item).

`byItem`: This is a logical argument. If `byItem=TRUE`, the transformation is applied on the columns of `x`. In the default `byItem=FALSE`, the function "picks" items row-wise.

In the default parameter settings of the function `pick()`, the parameters `k` and `cutoff` respectively are equal to `NULL`. In this case, the mean from N responses is used as a person-specific cut-off value (if `byItem=FALSE`). When `byItem=TRUE` (with `k,cutoff` equal to `NULL`) then the item mean over all individuals is used as an item specific cut-off value. The parameters `k` and `cutoff` are responsible for different dichotomization processes and they cannot be used simultaneously, which means that only one of the two arguments can be different than `NULL`.

In the case in which the user chooses to transform the data assuming that persons are asked to pick exactly K out of N items, ties can occur. If x_i is a response vector subject to transformation, in which ties exist, then we select among the tied items at random.

Generally, dichotomization should be avoided since it could distort the data structure and lead into information loss. Models that take into account information different categories should be prefered over dichotomization for polytomous data.

Applications

In this section we provide examples of how to use MUDFOLD method on two datasets, which are provided with the **mudfold** package. The first application is from the field of psychometrics while the second example is a linguistic application.

The commands `install.packages("mudfold")` and `library(mudfold)` will download, install and load the **mudfold** package so it can be used. The command `set.seed(1)` will set the seed for reproducibility.

Loneliness data

In order to demonstrate the functionality of the **mudfold** package we re-analyze questionnaire data following the strategy suggested by Post et al. (2001). For this purpose, we use a unidimensional measurement scale for loneliness that follows the definitions of a Rasch scale and has been constructed by de Jong-Gierveld and Kamphuls (1985). De Jong-Gierveld loneliness scale consists of eleven items, five of which are positive and six are negative. The items in the loneliness scale are given below and the sign next to the items corresponds to the item content.

A:	There is always someone I can talk to about my day to day problems	+
B:	I miss having a really close friend	-
C:	I experience a general sense of emptiness	-
D:	There are plenty of people I can lean on in case of trouble	+
E:	I miss the pleasure of company of others	-
F:	I find my circle of friends and acquaintances too limited	-
G:	There are many people that I can count on completely	+
H:	There are enough people that I feel close to	+
I:	I miss having people around	-
J:	Often I feel rejected	-
K:	I can call on my friends whenever I need them	+

Each item in the scale has three possible levels of response, i.e. “no", “more or less", “yes" and dichotomization methods that involve item reverse coding have been proposed by De Jong and van Tilburg (1999). These methods as well as the determination of dimensionality of this scale have been under critical discussion. Following this discussion, Post et al.

(2001) reanalyzed the loneliness scale data obtained from the NESTOR study (Knipscheer et al., 1995) using MUDFOLD in a three step analysis routine.

Persons with missing responses are removed from the data ($n_{miss} = 69$). The dataset with the complete responses is included in the R package **mudfold** in R data format. List-wise deletion in this case yields identical results with MICE imputation. Following the routine suggested by Post et al. (2001) responses of each subject are dichotomized setting "yes" versus "no" and "more or less".

The threshold that is used for the main analysis has been determined on the basis of MUDFOLD scale analysis on datasets with different thresholds. Specifically, the data has been dichotomized using as thresholds the response, (i) "yes", (ii) "more or less", (iii) different thresholds per item where the response category "more or less" is collapsed with the smaller category between "yes" and "no". The results from this analysis showed that dichotomizing the data at the higher preference will yield the best unfolding measurement scale for loneliness.

Dichotomizing the data at "yes" is straightforward with the `pick()` function.

```
data("Loneliness")
dat <- pick(Loneliness, cutoff = 3)
```

In the first step of the analysis, we conduct a MUDFOLD scale search on the transformed binary responses of $n = 3987$ individuals on $N = 11$ items. The λ_1 parameter in the `mudfold()` function is set to $\lambda_1 = 0.1$ since the default value leads to a minimal scale of length three.

```
Lonelifit <- mudfold(dat, lambda1 = 0.1, nboot = 100, seed = 1)
```

The function takes about five minutes to run 100 bootstrap iterations. The resulting scale and its associated statistics can be obtained by summarizing the `Lonelifit` object.

```
loneliSummary <- summary(Lonelifit, boot = TRUE)
```

The MUDFOLD scale for the `Loneliness` data in its estimated rank order is:

```
loneliScale <- loneliSummary$ITEM_STATS$ITEM_DESCRIPTIVES$items
loneliScale
## "G" "H" "D" "K" "C" "E" "I" "F"
```

The scale has length eight, with the first four items positively formulated and the last four negatively formulated. Items `A`,`B`, and `J` are excluded from the scale. This is because some triples (with respect to the item rank order) that include these items have scalability coefficient H_{hjk} lower than λ_2. Statistics for the resulting MUDFOLD scale and each item explicitly can be accessed directly from the summary object `loneliSummary`. Scale statistics with their bootstrap uncertainty estimates can be obtained with the following command.

```
loneliSummary$SCALE_STATS[1:3, ]
##               value  perc_lower95CI perc_upper95CI boot(mean) boot(bias) boot(se) boot(iter)
## H(scale)      0.536          0.436          0.571      0.511     -0.025    0.031        100
## ISO(scale)    0.078          0.001          1.753      0.384      0.306    0.459        100
## MAX(scale)    0.000          0.000          2.400      0.381      0.381    0.683        100
```

The output above, in each row shows a scale statistic and its columns correspond to the bootstrap properties of this statistic. The H coefficient for the scale shows strong evidence towards unidimensionality ($H_{\text{total}}(s) \approx 0.54$, $se = 0.031$), the ISO statistic is low ($ISO_{\text{total}} \approx 0.08$, $se = 0.459$) denoting small amount of violations of the manifest unimodality, and the MAX statistic is zero ($se = 0.683$) meaning no violations of the stochastic ordering.

Scale diagnostics are given in Figure 1 and 2. Visual inspection if the maxima of the CAM rows are a nondecreasing function of the item ranks, violations of the local independence assumption, and the IRF for each item in the `Loneliness` unfolding scale can be obtained by using the `diagnostics()` function as shown below.

```
par(mfrow = c(1, 2))
# testing for local independence
diagnostics(Lonelifit, which = "LI")
# visual inspection of moving maxima
diagnostics(Lonelifit, which = "STAR")
par(mfrow=c(2,4))
# visual inspection for IRF unimodality
```

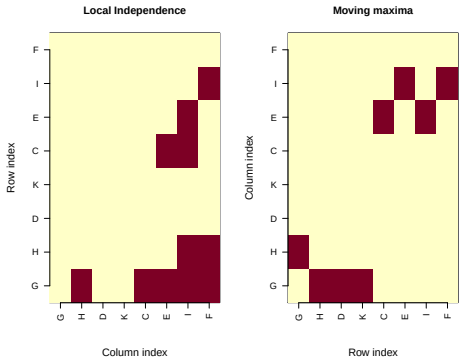

Figure 1: Left hand side: Red squares in the lower triangular part of the matrix represent pairs of conditionally dependent items. Right hand side: Red squares represent the position of the observed maxima in the CAM rows for the `Loneliness` unfolding scale.

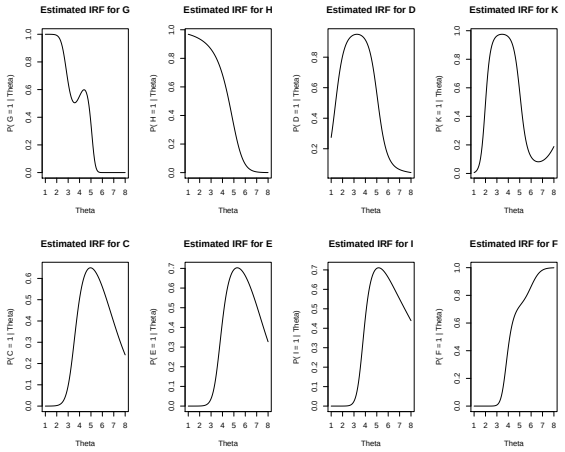

Figure 2: The estimated item response function for each item in the `Loneliness` unfolding scale.

```
diagnostics(Lonelifit, which = "UM")
par(mfrow = c(1, 1))
```

The H coefficients for each item in the scale are also available in the summary object and can be accessed by:

```
loneliSummary$ITEM_STATS$H_MUDFOLD_items
     value perc_lower95CI perc_upper95CI boot(mean) boot(bias) boot(se) boot(iter)
H(G)  0.54          0.444          0.573      0.510     -0.034    0.035         96
H(H)  0.52          0.440          0.543      0.495     -0.027    0.025         72
H(D)  0.51          0.400          0.553      0.498     -0.015    0.032         65
H(K)  0.51          0.440          0.554      0.495     -0.016    0.025         60
H(C)  0.55          0.404          0.590      0.513     -0.041    0.049         76
H(E)  0.57          0.491          0.610      0.555     -0.016    0.029         78
H(I)  0.55          0.493          0.586      0.541     -0.011    0.022         47
H(F)  0.52          0.349          0.546      0.464     -0.057    0.058         84
```

From the item fit we can see that the H coefficient for each item in the scale is above 0.5 which means that all the items are scalable together. Looking at the column `boot(iter)` of the output above you can get information for the number of times each item was included in a MUDFOLD scale out of $R = 100$ bootstrap iterations. The item `G` was the most frequently included item (96%) while the items `K,I` were included less frequently in a MUDFOLD scale compared to the other items (60% and 47% respectively). Typing `loneliSummary$ITEM_STATS$ISO_MUDFOLD_items` into the R console will return a summary of the ISO statistic for each item in the scale. The latter, shows that only small violations of unimodality occur for the items in the scale. The same holds for the MAX statistic (it can be accessed by `loneliSummary$ITEM_STATS$MAX_MUDFOLD_items`), which shows zero values for all the items in the scale.

After the scale is obtained and checked for its conformity to the unfolding principles we can visualize the estimated empirical IRFs and the distribution of the estimated person parameters. Plots for the IRFs and the person parameters can be obtained by:

```
plot(Lonelifit,plot.type = "IRF")
plot(Lonelifit,plot.type = "persons")
```

Figures 3 and 4 show the empirical estimates of the IRFs and the distribution of the person parameters respectively. In figure 3 you can see that the scale clearly consists of four positively formulated items in its beginning for which the IRF is decreasing as one moves from the left to the right of the scale, and four negatively formulated items in the end for which the IRF is increasing as one moves from the left to the right of the scale. In figure 4 we can see that the sample under consideration tends to feel less lonely since the distribution of the person parameters is skewed to the right. In such example, clearly any parametric model that assumed a latent normal distribution of the latent person parameters would be inappropriate.

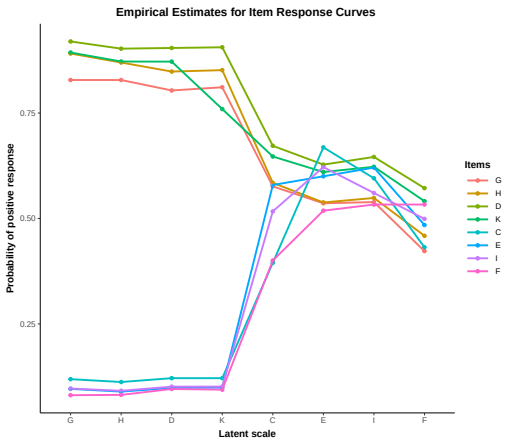

Figure 3: MUDFOLD's empirical estimates of the IRFs for the `Loneliness` unfolding scale.

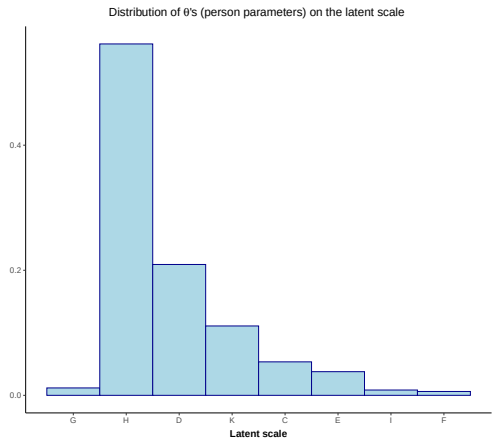

Figure 4: The distribution of the estimated person ranks for the `Loneliness` unfolding scale.

Plato's seven works data

In this section, we present an application of MUDFOLD method to the `Plato7` data set. This dataset is available from the R package **smacof** (de Leeuw and Mair, 2009) and has been also included in the **mudfold** package. The data can be loaded into the R environment with the command `data("Plato7")`.

`Plato7` contains information on the quantity distribution over the sentence ending from seven works of Plato (D. R. Cox, 1959). Specifically, the last five syllables from each sentence in seven Plato's works are extracted and categorized as short or long. This produces $2^5 = 32$ possible combinations of short-long syllables of length five, which are called clausulas and can be used to identify rhythmic changes in the literary style. The quantity of the clausulas in each work of Plato is recorded in terms of proportions.

The question is whether it is possible using these data to assign a chronological order to the works of Plato. Particularly, it is known that Plato wrote first the `Republic` and last the `Laws`. In between `Republic` and `Laws`, Plato wrote the `Critias`, `Philebus`, `Politicus`, `Sophist` and `Timaeus`. However, the exact order of these five works is unknown. Assuming that the change in Plato's literary style was monotone in time, we might be able to assign a time order in his works by analyzing the clausula's distribution in each Plato's work.

We consider the development of Plato's literary style as a unidimensional scale, on which clausulas and works are ordered. In this analysis we consider that the quantity of clausula i in Plato's work j will be governed by a proximity relation. That is, each clausula with a parameter θ_i on a latent literary style continuum tends to prefer (appear most frequently in) the works of Plato with parameters β_j close to θ_i.

Since the data is given in continuous form, we transform the percentages into binary format in order to apply MUDFOLD. We consider the mean quantity of each clausula as an explicit cut-off value for the transformation. The latter can be seen as a pick *any* out of N response process where the number of items "picked" varies across subjects. We can apply the transformation with the function `pick()` from the **mudfold** package in its default settings as follows.

```
dat.Plato <- pick(Plato7)
```

After the transformation, we end up with a matrix containing the binary preferences of $n = 32$ clausulas on $N = 7$ works of Plato. Now we can fit a MUDFOLD scale (with bootstrap for assessing parameter uncertainty) to the transformed data with the default search settings and study its summary.

```
fitPlato <- mudfold(dat.Plato, nboot = 100, seed = 1)
summaryPlato <- summary(fitPlato, boot = TRUE)
```

We can check the MUDFOLD scale from the summary object.

```
summaryPlato$SCALE_STATS[1:3, ]
##             value perc_lower95CI perc_upper95CI boot(mean) boot(bias) boot(se)  boot(iter)
## H(scale)   0.558          0.457          1.000      0.714      0.156    0.146        100
## ISO(scale) 0.146          0.000          1.129      0.141     -0.005    0.254        100
## MAX(scale) 0.000          0.000          0.850      0.035      0.035    0.191        100
```

The scale shows strong scalability properties with $H_{\text{total}(s)} = 0.56$, and low ISO statistic ($ISO_{\text{total}} = 0.15$). Since the scale is strong, the next step is to check the rank order of the items in the MUDFOLD scale and their scalability properties.

```
summaryPlato$ITEM_STATS$H_MUDFOLD_items
##               value perc_lower95CI perc_upper95CI boot(mean) boot(bias) boot(se) boot(iter)
## H(Republic)   0.66          0.362              1      0.761      0.097    0.179         70
## H(Sophist)    0.41          0.381              1      0.656      0.241    0.157         70
## H(Politicus)  0.58          0.392              1      0.662      0.078    0.161         62
## H(Philebus)   0.63          0.429              1      0.726      0.094    0.141         83
## H(Laws)       0.51          0.394              1      0.688      0.176    0.148         77
```

The results shows that the MUDFOLD scale has length five and the items `Critias` and `Timaeus` have been excluded from the measurement process. `Republic` is correctly ordered first and `Laws` is correctly ordered last among Plato's works. Almost all the items are strong unfolding items with $H_j(s)$ higher than 0.5 which means that the items are scalable together in one dimension. The item `Sophist` shows moderate unfolding strength with the lowest item scalability coefficient (i.e. $H_j(s) = 0.41$) while the item `Republic` is the strongest unfolding item in the scale.

Since the ISO statistic for the scale is positive one may wants to check which items are responsible for the small amount of manifest unimodality violations that are observed. Assessing these violations for each item involves checking their ISO statistics.

```
summaryPlato$ITEM_STATS$ISO_MUDFOLD_items
##                value perc_lower95CI perc_upper95CI boot(mean) boot(bias) boot(se) boot(iter)
## ISO(Republic)  0.104             0          0.365      0.036     -0.068    0.076         74
## ISO(Sophist)   0.042             0          0.326      0.039     -0.003    0.078         70
## ISO(Politicus) 0.000             0          0.082      0.005      0.005    0.018         65
## ISO(Philebus)  0.000             0          0.576      0.027      0.027    0.117         87
## ISO(Laws)      0.000             0          0.186      0.019      0.019    0.067         78
```

The obtained summary output for the ISO statistics of the items in the MUDFOLD scale show that `Republic` is the item with the higher manifest unimodality errors in its estimated IRF with an iso statistic value of 0.1. The higher uncertainty is observed for the item `Philebus` that shows a bootstrap standard error of 0.1.

The estimated empirical IRFs and the estimated IRFs for the items in the `Plato7` unfolding scale can be visualized with

```
plot(fitPlato, plot.type = "IRF")
par(mfrow = c(2, 3))
diagnostics(fitPlato, which = "UM")
par(mfrow = c(1, 1))
```

and the output is shown in figures 5 and 6 respectively. From figure 5 it can be seen that the scale consists of two items in the first positions (i.e. `Republic` and `Sophist`) with decreasing empirical IRFs as one moves from the left to the right hand side of the latent scale. These two items show small amount of manifest unimodality violations which can be seen at the end of their IRFs where the value of the curves is larger for the item `Laws` compared to item `Philebus`. Third in the scale is the item `Politicus` for which the empirical IRF shows a single-peak shape. `Politicus` is followed by the items `Philebus` and `Laws` with increasing empirical IRFs at positions four and five of the scale. The estimates of the IRFs are shown in figure 6 with no obvious violations of the IRF unimodality.

Other diagnostics can be obtained by the `diagnostics()` function. In this example the bootstrap estimate of the scale with the estimated MUDFOLD scale are slightly different. In such instances an additional element with a summary of the scale estimated by the bootstrap is included in the output. Accessing the summary of the bootstrap scale is straightforward with `summaryPlato$BOOT_SCALE`.

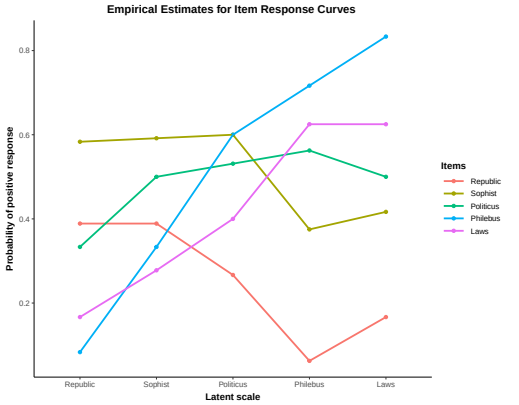

Figure 5: MUDFOLD's empirical estimates of the IRFs for the items in the `Plato7` unfolding scale.

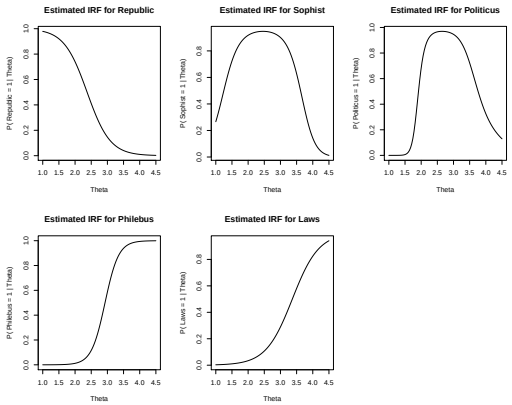

Figure 6: MUDFOLD's estimates of the IRFs for the items in the `Plato7` unfolding scale.

Summary

In this paper we introduced an R package named **mudfold** (Balafas et al., 2019). The latter is available under general public license (GPL $\geq$ 2) from the Comprehensive R Archive Network (CRAN) at `http://CRAN.R-project.org/package=mudfold`. This package implements a nonparametric item response theory model for unfolding proposed by Van Schuur (1984, 1988) and further developed by Post (1992) (see also Johnson (2006)).

The **mudfold** package is an addition to a broad family of R packages that fit IRT models. The approach described here is an additional exploratory and validation method when fitting such models. Moreover it adds to the package **mokken** for the case in which proximity item response data needs to be analysed.

Looking to the future our focus will be on extending the functionality of this package. In detail, we aim on the implementation of a more efficient item selection algorithm which can reduce the computational cost implied from the old fashioned iterative algorithm presented here when the sample size and item number are significantly increasing. Methodologies for handling multicategory type of items (Van Schuur, 1984) are not yet implemented in the package, however, we plan to extend its applicability in the future. Last but not least, a parametric version of MUDFOLD method based on the IRF implemented in the `mudfoldsim()` will offer a complete framework for the analysis of data that have been generated under an unfolding response process.

Bibliography

D. Andrich. The application of an unfolding model of the pirt type to the measurement of attitude. *Applied psychological measurement*, 12(1):33–51, 1988. URL `https://doi.org/10.1177/014662168801200105`. [p4, 15]

D. Andrich. A hyperbolic cosine irt model for unfolding direct responses of persons to items. In W. J. van der Linden and R. K. Hambleton, editors, *Handbook of Modern Item Response Theory*, pages 399–414. Springer New York, New York, NY, 1997. ISBN 978-1-4757-2691-6. URL `https://doi.org/10.1007/978-1-4757-2691-6_23`. [p1]

D. Andrich and G. Luo. A hyperbolic cosine latent trait model for unfolding dichotomous single-stimulus responses. *Applied Psychological Measurement*, 17(3):253–276, 1993. URL `https://doi.org/10.1177/014662169301700307`. [p1]

S. Balafas, W. Krijnen, and E. Wit. *mudfold: Multiple UniDimensional unFOLDing*, 2019. URL `https://CRAN.R-project.org/package=mudfold`. R package version 1.1.2. [p1, 22]

M. Boukes and H. G. Boomgaarden. Soft news with hard consequences? introducing a nuanced measure of soft versus hard news exposure and its relationship with political

cynicism. *Communication Research*, 42(5):701–731, 2015. URL https://doi.org/10.1177/0093650214537520. [p2]

S. V. Buuren, J. P. Brand, C. G. Groothuis-Oudshoorn, and D. B. Rubin. Fully conditional specification in multivariate imputation. *Journal of Statistical Computation and Simulation*, 76(12):1049–1064, 2006. URL https://doi.org/10.1080/10629360600810434. [p2]

A. Canty and B. D. Ripley. *boot: Bootstrap R (S-Plus) Functions*, 2017. R package version 1.3-20. [p2, 10]

R. P. Chalmers. mirt: A multidimensional item response theory package for the R environment. *Journal of Statistical Software*, 48(6):1–29, 2012. URL https://doi.org/10.18637/jss.v048.i06. [p2]

O. Chernyshenko, S. Stark, F. Drasgow, and B. Roberts. Constructing personality scales under the assumptions of an ideal point response process: Toward increasing the flexibility of personality measures. *Psychological assessment*, 19:88–106, 04 2007. URL https://doi.org/10.1037/1040-3590.19.1.88. [p1]

Y.-J. Choi and A. Asilkalkan. R packages for item response theory analysis: Descriptions and features. *Measurement: Interdisciplinary Research and Perspectives*, 17(3):168–175, 2019. URL https://doi.org/10.1080/15366367.2019.1586404. [p2]

C. H. Coombs. *A Theory Of Data*. Wiley, 1964. ISBN 978-0471171140. [p1, 4]

L. B. D. R. Cox. On a discriminatory problem connected with the works of Plato. *Journal of the Royal Statistical Society. Series B (Methodological)*, 21(1):195–200, 1959. ISSN 00359246. URL https://doi.org/10.1111/j.2517-6161.1959.tb00329.x. [p20]

G. J. De Jong and T. van Tilburg. Manual of the loneliness scale. *Amsterdam: VU University Amsterdam*, 1999. [p17]

J. de Jong-Gierveld and F. Kamphuls. The development of a rasch-type loneliness scale. *Applied psychological measurement*, 9(3):289–299, 1985. URL https://doi.org/10.1177/014662168500900307. [p17]

J. de Leeuw and P. Mair. Multidimensional scaling using majorization: SMACOF in R. *Journal of Statistical Software*, 31(3):1–30, 2009. URL https://doi.org/10.18637/jss.v031.i03. [p20]

B. Efron et al. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1): 1–26, 1979. URL https://doi.org/10.1214/aos/1176344455. [p2, 10]

T. R. Finserås, S. Pallesen, R. A. Mentzoni, E. Krossbakken, D. L. King, and H. Molde. Evaluating an internet gaming disorder scale using mokken scaling analysis. *Frontiers in Psychology*, 10:911, 2019. ISSN 1664-1078. URL https://doi.org/10.3389/fpsyg.2019.00911. [p1]

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010. URL https://doi.org/10.18637/jss.v033.i01. [p8]

L. Guttman. A basis for scaling qualitative data. *American Sociological Review*, 9(2):139–150, 1944. ISSN 00031224. URL https://doi.org/10.2307/2086306. [p4]

R. Hänggli. *Role of Dialogue in Public Opinion Formation*, pages 187–222. Springer International Publishing, Cham, 2020. ISBN 978-3-030-26582-3. URL https://doi.org/10.1007/978-3-030-26582-3_8. [p1]

H. Hoijtink. The measurement of latent traits by proximity items. *Applied psychological measurement*, 15(2):153–169, 1991. URL https://doi.org/10.1177/014662169101500205. [p1]

H. Hoijtink. Item response models for nonmonotone items. In K. Kempf-Leonard, editor, *Encyclopedia of Social Measurement*, pages 373 – 378. Elsevier, New York, 2005. ISBN 978-0-12-369398-3. URL https://doi.org/10.1016/B0-12-369398-5/00464-3. [p1]

M. S. Johnson. Nonparametric estimation of item and respondent locations from unfolding-type items. *Psychometrika*, 71(2):257–279, 2006. URL https://doi.org/10.1007/s11336-003-1098-9. [p2, 3, 11, 12, 22]

M. S. Johnson and B. W. Junker. Using data augmentation and markov chain monte carlo for the estimation of unfolding response models. *Journal of Educational and Behavioral Statistics*, 28(3):195–230, 2003. URL https://doi.org/10.3102/10769986028003195. [p1]

S. Karlin. *Total positivity*, volume 1. Stanford University Press, 1968. ISBN 978-0804703147. [p3]

C. P. Knipscheer, J. d. Jong-Gierveld, T. G. van Tilburg, P. A. Dykstra, et al. Living arrangements and social networks of older adults. *Amsterdam: VU University Amsterdam*, 1995. [p18]

P. Lee, S.-H. Joo, S. Stark, and O. S. Chernyshenko. Ggum-rank statement and person parameter estimation with multidimensional forced choice triplets. *Applied Psychological Measurement*, 43(3):226–240, 2019. URL https://doi.org/10.1177/0146621618768294. [p1]

R. J. Little and D. B. Rubin. Statistical analysis with missing data. *New York: Wiley, 1987*, 1987. URL https://doi.org/10.1002/9781119013563. [p11]

C.-W. Liu and W.-C. Wang. A general unfolding irt model for multiple response styles. *Applied Psychological Measurement*, 43(3):195–210, 2019. URL https://doi.org/10.1177/0146621618762743. [p1]

J. Loevinger. The technic of homogeneous tests compared with some aspects of "scale analysis" and factor analysis. *Psychological bulletin*, 45(6):507, 1948. URL https://doi.org/10.1037/h0055827. [p2, 4]

G. Luo. A class of probabilistic unfolding models for polytomous responses. *Journal of Mathematical Psychology*, 45(2):224 – 248, 2001. ISSN 0022-2496. URL https://doi.org/10.1006/jmps.2000.1310. [p1]

G. Luo, D. Andrich, and I. Styles. The jml estimation of the generalised unfolding model incorporating the latitude of acceptance parameter. *Australian Journal of Psychology*, 50(3): 187–198, 1998. URL https://doi.org/10.1080/00049539808258795. [p1]

M. D. Maraun and N. T. Rossi. The extra-factor phenomenon revisited: Unidimensional unfolding as quadratic factor analysis. *Applied Psychological Measurement*, 25(1):77–87, 2001. URL https://doi.org/10.1177/01466216010251006. [p1]

A. Maydeu-Olivares, A. Hernández, and R. P. McDonald. A multidimensional ideal point item response theory model for binary data. *Multivariate Behavioral Research*, 41(4):445–472, 2006. URL https://doi.org/10.1207/s15327906mbr4104_2. PMID: 26794914. [p1]

R. J. Mokken. *A theory and procedure of scale analysis: With applications in political research*, volume 1. Walter de Gruyter, 1971. ISBN 978-3-11-081320-3. [p2, 4]

R. J. Mokken. Nonparametric models for dichotomous responses. In W. J. van der Linden and R. K. Hambleton, editors, *Handbook of Modern Item Response Theory*, pages 351–367. Springer New York, New York, NY, 1997. ISBN 978-1-4757-2691-6. URL https://doi.org/10.1007/978-1-4757-2691-6_20. [p3]

Y. Noel. A beta unfolding model for continuous bounded responses. *Psychometrika*, 79(4): 647–674, Oct 2014. ISSN 1860-0980. URL https://doi.org/10.1007/s11336-013-9361-1. [p1]

W. J. Post. *Nonparametric Unfolding Models: A Latent Structure Approach*. M & T series. DSWO Press, 1992. ISBN 978-9066950641. [p1, 2, 3, 8, 9, 22]

W. J. Post and T. A. Snijders. Nonparametric unfolding models for dichotomous data. *Methodika*, 1993. [p1, 2, 3, 8]

W. J. Post, M. A. van Duijn, and B. van Baarsen. Single-peaked or monotone tracelines? on the choice of an irt model for scaling data. In *Essays on item response theory*, pages 391–414. Springer, 2001. URL https://doi.org/10.1007/978-1-4613-0169-1_21. [p17, 18]

G. Rasch. On general laws and the meaning of measurement in psychology. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 4: Contributions to Biology and Problems of Medicine*, pages 321–333, Berkeley, Calif., 1961. University of California Press. URL http://projecteuclid.org/euclid.bsmsp/1200512895. [p1]

W. D. Rinkel, M. H. Aziz, J. W. Van Neck, M. C. Cabezas, L. A. van der Ark, and J. H. Coert. Development of grading scales of pedal sensory loss using mokken scale analysis on the rotterdam diabetic foot study test battery data. *Muscle & Nerve*, 60(5):520–527, 2019. URL https://doi.org/10.1002/mus.26628. [p1]

J. S. Roberts and J. E. Laughlin. A unidimensional item response model for unfolding responses from a graded disagree-agree response scale. *Applied Psychological Measurement*, 20(3):231–255, 1996. URL https://doi.org/10.1177/014662169602000305. [p1, 2]

J. S. Roberts and V. M. Thompson. Marginal maximum a posteriori item parameter estimation for the generalized graded unfolding model. *Applied Psychological Measurement*, 35(4): 259–279, 2011. URL https://doi.org/10.1177/0146621610392565. [p1]

J. S. Roberts, J. R. Donoghue, and J. E. Laughlin. A general item response theory model for unfolding unidimensional polytomous responses. *Applied Psychological Measurement*, 24 (1):3–32, 2000. URL https://doi.org/10.1177/01466216000241001. [p1, 2]

J. S. Roberts, H.-r. Fang, W. Cui, and Y. Wang. Ggum2004: A windows-based program to estimate parameters in the generalized graded unfolding model. *Applied Psychological Measurement*, 2006. URL https://doi.org/10.1177/0146621605280141. [p2]

D. B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976. URL https://doi.org/10.1093/biomet/63.3.581. [p11]

K. Sijtsma. Nonparametric item response theory models. In K. Kempf-Leonard, editor, *Encyclopedia of Social Measurement*, pages 875 – 882. Elsevier, New York, 2005. ISBN 978-0-12-369398-3. URL https://doi.org/10.1016/B0-12-369398-5/00459-X. [p1]

K. Sijtsma and B. W. Junker. Item response theory: Past performance, present developments, and future expectations. *Behaviormetrika*, 33(1):75–102, 2006. URL https://doi.org/10.2333/bhmk.33.75. [p1]

S. Stark, O. Chernyshenko, F. Drasgow, and B. Williams. Examining assumptions about item responding in personality assessment: Should ideal point methods be considered for scale development and scoring? *Journal of Applied Psychology*, 91(1):25–39, 2006. ISSN 0021-9010. URL https://doi.org/10.1037/0021-9010.91.1.25. [p1]

I. Sulis and M. Porcu. Handling missing data in item response theory. assessing the accuracy of a multiple imputation procedure based on latent class analysis. *Journal of Classification*, 34(2):327–359, Jul 2017. ISSN 1432-1343. URL https://doi.org/10.1007/s00357-017-9220-3. [p12]

J. N. Tendeiro and S. Castro-Alvarez. *GGUM: Generalized Graded Unfolding Model*, 2018. URL https://CRAN.R-project.org/package=GGUM. R package version 0.3.3. [p2]

L. L. Thurstone. A law of comparative judgment. *Psychological review*, 34(4):273, 1927. URL https://doi.org/10.1037/h0070288. [p1, 11]

L. L. Thurstone. Attitudes can be measured. *American journal of Sociology*, pages 529–554, 1928. URL https://doi.org/10.1086/214483. [p1, 11]

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996. URL https://doi.org/10.1111/j.2517-6161.1996.tb02080.x. [p8]

S. van Buuren and K. Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45(3):1–67, 2011. URL https://doi.org/10.18637/jss.v045.i03. [p2]

L. A. Van der Ark. Mokken scale analysis in R. *Journal of Statistical Software*, 20(11):1–19, 2007. URL https://doi.org/10.18637/jss.v020.i11. [p2]

L. A. Van der Ark. New developments in mokken scale analysis in R. *Journal of Statistical Software*, 48(5):1–27, 2012. URL https://doi.org/10.18637/jss.v048.i05. [p2]

W. Van Schuur. Stochastic unfolding. In *Sociometric research*, pages 137–158. Springer, 1988. URL https://doi.org/10.1007/978-1-349-19051-5_9. [p2, 11, 12, 22]

W. H. Van Schuur. *Structure in Political Beliefs: A New Model for Stochastic Unfolding with Application to European Party Activities*. CT Press, 1984. ISBN 978-9070758042. [p1, 2, 22]

W. H. Van Schuur. Nonparametric unidimensional unfolding for multicategory data. *Political Analysis*, 4:41–74, 1992. URL https://doi.org/10.1093/pan/4.1.41. [p2]

G. R. Warnes, B. Bolker, and T. Lumley. *gtools: Various R Programming Tools*, 2015. URL https://CRAN.R-project.org/package=gtools. R package version 3.5.0. [p6]

H. Wickham. Reshaping data with the reshape package. *Journal of Statistical Software*, 21(12):1–20, 2007. URL https://doi.org/10.18637/jss.v021.i12. [p14]

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009. ISBN 978-0-387-98140-6. URL https://doi.org/10.1007/978-0-387-98141-3. [p14]

S. Wood. *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC, 2 edition, 2017. [p9]

S. N. Wood. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society (B)*, 73(1):3–36, 2011. URL https://doi.org/10.1111/j.1467-9868.2010.00749.x. [p9]

A. Zeileis and G. Grothendieck. zoo: S3 infrastructure for regular and irregular time series. *Journal of Statistical Software*, 14(6):1–27, 2005. URL https://doi.org/10.1007/978-0-387-98141-3. [p14]

Spyros E. Balafas
Bernoulli Institute for Mathematics, Computer Science & Artificial Intelligence
University of Groningen (RUG)
Bernoulliborg, Rm. 460, Nijenborgh 9
9747 AG Groningen
The Netherlands
s.balafas@rug.nl

Wim P. Krijnen
Bernoulli Institute for Mathematics, Computer Science & Artificial Intelligence
University of Groningen (RUG)
Bernoulliborg, Nijenborgh 9
9747 AG Groningen
The Netherlands
w.p.krijnen@rug.nl

Wendy J. Post
Orthopedagogy & Clinical Educational Science
University of Groningen (RUG)
Grote Rozenstraat 38
9712 TJ Groningen
The Netherlands
w.j.post@rug.nl

Ernst C. Wit
Faculty of Science & Informatics
Universita della Svizzera Italiana (USI)
Via Buffi 13
6900 Lugano
Switzerland
wite@usi.ch