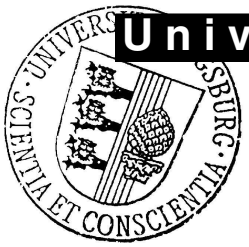



Universität Augsburg

Institut für
Mathematik

Anatol Sargin, Ali Ünlü

# DAKS: An R Package for Data Analysis Methods in Knowledge Space Theory

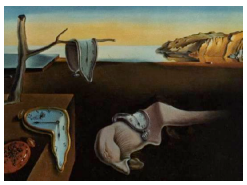
Preprint Nr. 08/2009 — 17. April 2009
Institut für Mathematik, Universitätsstraße, D-86135 Augsburg http://www.math.uni-augsburg.de/

**Impressum:**

*Herausgeber:*
Institut für Mathematik
Universität Augsburg
86135 Augsburg
`http://www.math.uni-augsburg.de/forschung/preprint/`

*ViSdP:*
Ali Ünlü
Institut für Mathematik
Universität Augsburg
86135 Augsburg

*Preprint:* Sämtliche Rechte verbleiben den Autoren © 2009



*Journal of Statistical Software*

*MMMMMM YYYY, Volume VV, Issue II.* *http://www.jstatsoft.org/*

# DAKS: An R Package for Data Analysis Methods in Knowledge Space Theory

**Anatol Sargin**
University of Augsburg

**Ali Ünlü**
University of Augsburg

**Abstract**

Knowledge space theory is part of psychometrics and provides a theoretical framework for the modeling, assessment, and training of knowledge. It utilizes the idea that some pieces of knowledge may imply others, and is based on order and set theory. We introduce the R package **DAKS** for performing basic and complex operations in knowledge space theory. This package implements three inductive item tree analysis algorithms for deriving quasi orders from binary data, the original, corrected, and minimized corrected algorithms. It provides functions for computing population and estimated asymptotic variances of the *diff* fit measures, and for switching between test item and knowledge state representations. Other features are a Hasse diagram drawing device, a data simulation tool based on a finite mixture latent variable model, and a function for computing response pattern and knowledge state frequencies. We describe the functions of the package and demonstrate their usage by real and simulated data examples.

*Keywords*: knowledge space theory, psychometrics, exploratory data analysis, R.

## 1. Introduction

Knowledge space theory (KST) was introduced by Doignon and Falmagne (1985). Most of the theory is presented in a monograph by Doignon and Falmagne (1999); for applications see Albert and Lukas (1999), and for survey articles see Doignon and Falmagne (1987), Falmagne (1989), and Falmagne, Koppen, Villano, Doignon, and Johannesen (1990). KST provides a theoretical framework for the modeling, assessment, and training of knowledge. This theory utilizes the idea that some pieces of knowledge may imply others. For instance, the mastery of a test question may imply the mastery of other test questions. Implications between pieces of knowledge are modeled in KST by order and set theoretic structures. Based on such a framework, KST has been successfully applied for computerized adaptive assessment and training; for example, see the ALEKS system (`http://www.aleks.com/`), a Web-based,

artificially intelligent assessment and learning system.

Deriving implications from data plays an important role in KST. Three inductive item tree analysis (IITA) algorithms have been proposed for deriving implications from dichotomous data: the original IITA algorithm (Schrepp 2003), and the corrected and minimized corrected IITA algorithms (Sargin and Ünlü 2009; Ünlü and Sargin 2008a). These methods constitute the main part of the package **DAKS**. Currently available software implementing the original IITA algorithm is **ITA 2.0** by Schrepp (2006). Compared to this stand-alone software that runs only on Windows, the package **DAKS** is embedded in the comprehensive R computing environment and provides much more functionalities such as more flexible input/output features.

R (R Development Core Team 2006, `http://www.r-project.org/`) is a language and environment for statistical computing and graphics. It gives users the possibility to include own software packages for handling specific tasks. Besides the three IITA algorithms, the package **DAKS** implements functions for computing population and estimated asymptotic variances of the fit measures, and for switching between test item and knowledge state representations. Other features are a Hasse diagram drawing device, a data simulation tool, and a function for computing response pattern and knowledge state frequencies.

In Section 2, the basic deterministic and probabilistic concepts of KST and the three IITA algorithms are reviewed. In Section 3, the package **DAKS** is presented and its functions are explained. In Section 4, the package **DAKS** is demonstrated using real and simulated data.

## 2. Knowledge space theory and data analysis methods

We briefly recapitulate the basic concepts of KST relevant for this work and the three IITA algorithms. Details can be found in the respective references afore mentioned.

### 2.1. Basic concepts of knowledge space theory

Assume a set $Q$ of $m$ dichotomous items. Mastering an item $j \in Q$ may imply mastering another item $i \in Q$. If no response errors are made, these implications, $j \rightarrow i$, entail that only certain response patterns (represented by subsets of $Q$) are possible. Those response patterns are called knowledge states, and the set of all knowledge states (including $\emptyset$ and $Q$) is called a knowledge structure, and denoted by $\mathcal{K}$. The knowledge structure $\mathcal{K}$ is a subset of $2^Q$, the power set of $Q$. Implications are assumed to form a quasi order, that is, a reflexive, transitive binary relation, $\sqsubseteq$ on the item set $Q$. In other words, an implication $j \rightarrow i$ stands for the pair $(i, j) \in \sqsubseteq$, also denoted by $i \sqsubseteq j$. Quasi orders are referred to as surmise relations in KST.

A possible application is an aptitude test, where participants can solve (coded 1) or fail to solve (coded 0) a question. In this paper, the latter interpretation is used to illustrate the IITA algorithms.

Implications are latent and not directly observable, due to random response errors. A person who is actually unable to solve an item, but does so, makes a lucky guess. On the other hand, a person makes a careless error, if he fails to solve an item which he masters. A probabilistic extension of the knowledge structure model covering random response errors is the basic local independence model. In Section 4.2, we use that probability model for simulating the data.

A quadruple $(Q, \mathcal{K}, p, r)$ is called a basic local independence model (BLIM) if and only if

1. $(Q, \mathcal{K})$ is a knowledge structure,
2. $p$ is a probability distribution on $\mathcal{K}$, that is, $p : \mathcal{K} \to ]0, 1[\,, K \mapsto p(K)$, with $p(K) > 0$ for any $K \in \mathcal{K}$, and $\sum_{K \in \mathcal{K}} p(K) = 1$,
3. $r$ is a response function for $(Q, \mathcal{K}, p)$, that is, $r : 2^Q \times \mathcal{K} \to [0, 1]$, $(R, K) \mapsto r(R, K)$, with $r(R, K) \geq 0$ for any $R \in 2^Q$ and $K \in \mathcal{K}$, and $\sum_{R \in 2^Q} r(R, K) = 1$ for any $K \in \mathcal{K}$,
4. $r$ satisfies local independence, that is,

$$r(R, K) = \prod_{q \in K \setminus R} \beta_q \cdot \prod_{q \in K \cap R} (1 - \beta_q) \cdot \prod_{q \in R \setminus K} \eta_q \cdot \prod_{q \in Q \setminus (R \cup K)} (1 - \eta_q),$$

with two constants $\beta_q, \eta_q \in [0, 1[$ for each $q \in Q$, respectively called careless error and lucky guess probabilities at $q$.

Here, $K \setminus R := \{q \in Q : q \in K \text{ and } q \notin R\}$, $K \cap R := \{q \in Q : q \in K \text{ and } q \in R\}$, $R \setminus K := \{q \in Q : q \in R \text{ and } q \notin K\}$, and $Q \setminus (R \cup K) := \{q \in Q : q \notin R \text{ and } q \notin K\}$. The items in $K \setminus R$, $K \cap R$, $R \setminus K$, and $Q \setminus (R \cup K)$ are mastered but not solved (careless error), mastered and solved (no careless error), solved but not mastered (lucky guess), and not solved and not mastered (no lucky guess), respectively.

Let $n$ be the sample size. The data are the observed absolute counts of response patterns $R \subset Q$. Let $D$ denote the corresponding $n \times m$ data matrix of 0/1 item scores. The data are assumed to be multinomially distributed over $2^Q$. Let $\rho(R)$ denote the (unknown) true probability of occurrence of a response pattern $R$. The BLIM is based on the following assumptions. To each knowledge state $K \in \mathcal{K}$ is attached a probability $p(K)$ measuring the likelihood that a respondent is in state $K$. For a manifest response pattern $R \subset Q$ and a latent knowledge state $K \in \mathcal{K}$, $r(R, K)$ specifies the conditional probability of response pattern $R$ for a respondent in state $K$. The item responses of a respondent are assumed to be independent given the knowledge state of the respondent (local independence). The response error, that is, careless error and lucky guess, probabilities $\beta_q$ and $\eta_q$ are attached to the items and do not vary with the knowledge states.

The BLIM allows expressing the occurrence probabilities $\rho(R)$ of response patterns $R$ by means of the model parameters $p(K)$ and $\beta_q, \eta_q$:

$$\begin{aligned} \rho(R) \quad = \quad \sum_{K \in \mathcal{K}} \Bigg\{ & \left[ \prod_{q \in K \setminus R} \beta_q \right] \cdot \left[ \prod_{q \in K \cap R} (1 - \beta_q) \right] \\ & \cdot \left[ \prod_{q \in R \setminus K} \eta_q \right] \cdot \left[ \prod_{q \in Q \setminus (R \cup K)} (1 - \eta_q) \right] \Bigg\} p(K). \end{aligned}$$

The BLIM is a restricted latent class model (see, e.g., Ünlü 2006). The number of independent model parameters is $2|Q| + (|\mathcal{K}| - 1)$, where $|\mathcal{K}|$ denotes the size of $\mathcal{K}$. Since $|\mathcal{K}|$ generally tends to be prohibitively large in practice, parameter estimation and model testing based on classical maximum likelihood methodology are not feasible in general. This is why exploratory methods such as the IITA algorithms are important in KST.

A knowledge structure closed under union and intersection is called a quasi ordinal knowledge space. Quasi ordinal knowledge spaces and surmise relations are equivalent formulations.

According to the Birkhoff (1937) theorem, there exists a one-to-one correspondence between the collection of all quasi ordinal knowledge spaces $\mathcal{K}$ on a domain $Q$, and the collection of all surmise relations $\sqsubseteq$ on $Q$. Such a correspondence is defined through the two equivalences:

$$\begin{aligned} p \sqsubseteq q \quad &:\Longleftrightarrow \quad [\forall K \in \mathcal{K} : \{q \in K \Longrightarrow p \in K\}]\,, \\ K \in \mathcal{K} \quad &:\Longleftrightarrow \quad [\forall (p \sqsubseteq q) : \{q \in K \Longrightarrow p \in K\}]\,. \end{aligned}$$

This theorem is important from a practical point of view. Though the quasi ordinal knowledge space and surmise relation models are empirically interpreted at the different levels of persons and items, they are connected with each other mathematically, through Birkhoff's theorem. This theorem is realized in the package **DAKS** using two functions for switching between test item and knowledge state representations (see Section 3.2).

## 2.2. Inductive item tree analysis algorithms

The three IITA algorithms are exploratory methods for extracting surmise relations from data. In each algorithm, competing binary relations are generated, and a fit measure is computed for every relation in order to find the quasi order that fits the data best. In the following, the algorithms are briefly reviewed.

For the original IITA version (Schrepp 2003) the algorithm is:

1. For two items $i, j$, the value $b_{ij} := |\{R \in D | i \notin R \wedge j \in R\}|$ is the number of counterexamples, that is, the number of observed response patterns in the data matrix $D$ contradicting $j \rightarrow i$. Based on these values, binary relations $\sqsubseteq_L$ for $L = 0, ..., n$ are defined. Let $i \sqsubseteq_0 j :\Leftrightarrow b_{ij} = 0$. The relation $\sqsubseteq_0$ is a quasi order. Construct inductively: Assume $\sqsubseteq_L$ is transitive. Define $S^{(0)}_{L+1} := \{(i,j) | b_{ij} \leq L + 1 \wedge i \not\sqsubseteq_L j\}$. From $S^{(0)}_{L+1}$, exclude those item pairs that cause an intransitivity in $\sqsubseteq_L \cup S^{(0)}_{L+1}$; the remaining pairs are referred to as $S^{(1)}_{L+1}$. This process continues iteratively, $k$ times, until no intransitivity is caused. The generated relation $\sqsubseteq_{L+1} := \sqsubseteq_L \cup S^{(k)}_{L+1}$ is a quasi order by construction.

2. The coefficient $\mathit{diff}_o(\sqsubseteq_L, D)$ is used to assess the fit of each quasi order $\sqsubseteq_L$ to the binary data matrix $D$ (see below).

3. Choose the quasi order with minimum $\mathit{diff}_o(\sqsubseteq_L, D)$ value.

For the corrected and minimized corrected IITA versions (Sargin and Ünlü 2009) the algorithms are:

1. The generation of the selection set of quasi orders is the same as in the original IITA version.

2. The coefficients $\mathit{diff}_c(\sqsubseteq_L, D)$ and $\mathit{diff}_{mc}(\sqsubseteq_L, D)$ are used to assess the fit of each quasi order $\sqsubseteq_L$ to the binary data matrix $D$ (see below), respectively.

3. Choose the quasi orders with minimum $\mathit{diff}_c(\sqsubseteq_L, D)$ and $\mathit{diff}_{mc}(\sqsubseteq_L, D)$ values, respectively.

The *diff* fit measures $diff_o$, $diff_c$, and $diff_{mc}$ are defined by

$$diff\,(\sqsubseteq, D) = \frac{1}{m(m-1)} \sum_{i \neq j} (b_{ij} - b^*_{ij})^2,$$

where corresponding estimates $b^*_{ij}$ are used. These estimates are computed based on a single error probability.

In the original IITA version this single error rate is computed by

$$\gamma_{\sqsubseteq} = \frac{1}{|\sqsubseteq| - m} \sum_{i \sqsubseteq j, i \neq j} \frac{b_{ij}}{p_j n}.$$

If $(i, j) \in \sqsubseteq$, the expected number of counterexamples is estimated by $b^*_{ij} = \gamma_{\sqsubseteq} p_j n$. If $(i, j) \notin \sqsubseteq$, the estimate $b^*_{ij} = (1 - p_i) p_j n (1 - \gamma_{\sqsubseteq})$ is used.

Better estimators than those used in the original algorithm are proposed by Sargin and Ünlü (2009). In the corrected IITA version the same $\gamma_{\sqsubseteq}$ and $b^*_{ij} = \gamma_{\sqsubseteq} p_j n$ for $(i, j) \in \sqsubseteq$ are used. The choice for $b^*_{ij}$ in the case of $(i, j) \notin \sqsubseteq$ now depends on whether $(j, i) \notin \sqsubseteq$ or $(j, i) \in \sqsubseteq$. If $(i, j) \notin \sqsubseteq$ and $(j, i) \notin \sqsubseteq$, set $b^*_{ij} = (1 - p_i) p_j n$. If $(i, j) \notin \sqsubseteq$ and $(j, i) \in \sqsubseteq$, set $b^*_{ij} = (p_j - p_i + \gamma_{\sqsubseteq} p_i) n$.

In the minimized corrected IITA version the corrected estimators $b^*_{ij}$ as in the $diff_c$ coefficient are used. Minimizing the *diff* expression as a function of the error probability $\gamma_{\sqsubseteq}$ gives $\gamma_{\sqsubseteq} = -\frac{x_1 + x_2}{x_3 + x_4}$, where

$$\begin{aligned}
x_1 &= \sum_{i \not\sqsubseteq j \,\wedge\, j \sqsubseteq i} -2 b_{ij} p_i n + 2 p_i p_j n^2 - 2 p_i^2 n^2, \\
x_2 &= \sum_{i \sqsubseteq j} -2 b_{ij} p_j n, \\
x_3 &= \sum_{i \not\sqsubseteq j \,\wedge\, j \sqsubseteq i} 2 p_i^2 n^2, \\
x_4 &= \sum_{i \sqsubseteq j} 2 p_j^2 n^2.
\end{aligned}$$

The idea is to use the corrected estimators and to optimize the fit criterion. The fit measure then favors quasi orders that lead to smallest minimum discrepancies, or equivalently, largest maximum matches, between the observed and expected numbers of counterexamples.

In Ünlü and Sargin (2008a), we introduce the population analogs of the *diff* fit measures, interpret the coefficients as maximum likelihood estimators (MLEs) for the corresponding population values, and show for these estimators the quality properties asymptotic efficiency, asymptotic normality, asymptotic unbiasedness, and consistency. This is briefly reviewed.

Consider the transformed sample *diff* coefficients $diff := diff\,/n^2$. The division is necessary to cancel out sample size $n$ in replacements of sample quantities with population quantities. Given the multinomial probability distribution on the set of all response patterns, make the following replacements in the arguments, $b_{ij}$ and $p_i$, of the sample *diff* coefficients:

$$\frac{b_{ij}}{n} \;\rightarrow\; P(i = 0, j = 1) = \sum_{R \in 2^Q, i \notin R \,\wedge\, j \in R} \rho(R),$$

$$p_i \quad \rightarrow \quad P(i=1) = \sum_{R \in 2^Q, i \in R} \rho(R).$$

This gives three population *diff* coefficients corresponding to the sample *diff* coefficients.

The sample *diff* coefficients are the obvious sample analogs of these population fit measures. They are reobtained by replacing the arguments $\rho(R)$ of the population *diff* measures with the MLEs $n(R)/n$ of the multinomial distribution, where $n(R)$ are the absolute counts of response patterns $R \in 2^Q$. According to the invariance property of MLEs (e.g., Casella and Berger 2002), the sample *diff* coefficients are the MLEs for the corresponding population *diff* coefficients.

The MLE for the multinomial distribution fulfills required regularity conditions and hence is asymptotically efficient (e.g., Casella and Berger 2002). The population *diff* coefficients are continuous functions of the multinomial cell probabilities $\rho(R)$. Therefore the sample *diff* coefficients are asymptotically efficient, asymptotically normal, asymptotically unbiased, and consistent estimators for the population values.

The functions of the package **DAKS** realizing the IITA algorithms in sample and population quantities are described in Section 3.2, and their usage by real and simulated data examples is demonstrated in Section 4.

## 3. Implementation in the package DAKS

In this section, we describe how surmise relations and knowledge structures are implemented, and discuss the functions of this package.

### 3.1. Surmise relations and knowledge structures in DAKS

A quasi order is a set of tuples, where each tuple is a pair $(i, j)$ representing the implication $j \rightarrow i$. This is implemented in **DAKS** using the package **sets** (developed by David Meyer and Kurt Hornik). The latter, in combination with the package **relations** (developed by Kurt Hornik and David Meyer), are utilized in **DAKS**, because they provide useful functions for operating with surmise relations and knowledge structures. The following R output shows an example quasi order:

```
{(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)}
```

or

```
{(1L, 2L), (1L, 3L), (1L, 4L), (2L, 3L), (2L, 4L), (3L, 4L)}
```

This code is to be read: item 1 is implied by items 2, 3, and 4, item 2 is implied by items 3 and 4, and item 3 is implied by item 4. This gives the chain $4 \rightarrow 3 \rightarrow 2 \rightarrow 1$. Note that in the second code line an item $i$ is represented by $iL$. This transformation takes place internally in the packages **sets** or **relations**, but it does not have any influence. Both representations are equal:

```
R> 1 == 1L
[1] TRUE
```

Note that reflexive pairs are not shown in order to reveal implications between different items only, and to save computing time. Surmise relations always contain all reflexive pairs, and these are included whenever required by the package **DAKS**.

A knowledge structure is implemented as a binary matrix, where rows and columns stand for knowledge states and items, respectively. Each entry of the matrix, 1 or 0, represents mastering or not mastering an item in a corresponding state. The following R output shows the knowledge structure corresponding to the above quasi order:

```
     [,1] [,2] [,3] [,4]
[1,]    0    0    0    0
[2,]    1    0    0    0
[3,]    1    1    0    0
[4,]    1    1    1    0
[5,]    1    1    1    1
```

### 3.2. Functions of the package DAKS

The two functions for switching between test item and knowledge state representations (cf. Birkhoff's theorem in Section 2.1) are:

```
state2imp(P)
imp2state(imp, items)
```

The first function transforms a set of knowledge states (ought to be a quasi ordinal knowledge space) `P` to the corresponding set of implications (the surmise relation). Note that for any set of knowledge states the returned binary relation is a surmise relation. The number of items of the domain taken as basis for `P` is determined from the number of columns of the matrix `P`. The second function transforms a set of implications (ought to be a surmise relation) `imp` to the corresponding set of knowledge states (the quasi ordinal knowledge space). Note that for any set of implications the returned knowledge structure is a quasi ordinal knowledge space. The number of items of the domain taken as basis for `imp`, the argument `items`, must be specified explicitly; because some of the items may not be comparable with any other.

A function to compute the absolute frequencies of the occurring response patterns, and optionally, the absolute frequencies of a collection of knowledge states in a dataset is:

```
pattern(dataset, n = 5, P = NULL)
```

Argument `n` refers to response patterns. If `n` is specified, the response patterns with the `n` highest frequencies are returned (along with their frequencies). If `pattern` is called without specifying `n` explicitly, by default `n = 5` is used. If `n` is larger than the number of different response patterns in the `dataset`, `n` is set the number of different response patterns. The optional matrix `P` gives the knowledge states to be used; `pattern` then additionally returns information about how often the knowledge states occur in the `dataset`. The default `P = NULL` corresponds to no knowledge states being specified; `pattern` then only returns information about response patterns (as described previously).

A data simulation tool based on the BLIM (Section 2.1) is included in the package:

```
simu(items, size, ce, lg, imp = NULL, delta)
```

The number of response patterns to be simulated (the sample size) is specified by `size`, the careless error and lucky guess noise parameters are given by `ce` and `lg`, respectively. The single careless error `ce` and lucky guess `lg` probabilities are assumed to be constant over all items. (The general form of the BLIM allows for varying careless error and lucky guess rates from item to item, which is not identifiable in general, however.) The argument `items` gives the number of items of the domain taken as basis for the quasi order underlying the simulation. A specific underlying quasi order can be passed manually via `imp`, or it can be generated randomly. If a quasi order is specified manually, Birkhoff's theorem (Section 2.1) is used to derive the corresponding quasi ordinal knowledge space. The latter is equipped with the error probabilities `ce` and `lg` to give the BLIM that is used for simulating the data. If `imp = NULL`, the underlying quasi order is generated randomly as follows. All reflexive pairs are added to the relation. The constant `delta` is utilized as the probability for adding each of the remaining non-reflexive item pairs to the relation. The transitive closure of this relation is computed, and the resulting quasi order then is the surmise relation underlying the simulation.

This simulation tool returns the simulated binary dataset, and the surmise relation and its corresponding quasi ordinal knowledge space used for simulating the data. The probability specified by `delta` does not necessarily correspond to the portion of implications added to the randomly generated quasi order, because the transitive closure is formed. In Sargin and Ünlü (2009), a normal sampling scheme for drawing `delta` values is proposed. This sampling scheme provides far better representative samples of quasi orders than simply drawing `delta` values uniformly from the unit interval. (Surmise relations or knowledge structures, and the representativeness of samples of these, are very important in simulation studies investigating IITA type data analysis methods. The IITA algorithms are sensitive to the underlying surmise relation that is used, and to test their performances objectively a representative sample of the collection of all quasi orders is needed.)

Another basic function of the package **DAKS** is a Hasse diagram drawing device:

```
hasse(imp, items)
```

This function plots the Hasse diagram of a surmise relation `imp` (more precisely, of the corresponding quotient set) using the package **Rgraphviz** from Bioconductor (`http://www.bioconductor.org/`), which is an interface between R and Graphviz (Graph Visualization Software, `http://graphviz.org/`). Users must install Graphviz on their computers to plot such a diagram. The argument `items` gives the number of items of the domain taken as basis for `imp`. The function `hasse` cannot plot equally informative items. (Two items $i$ and $j$ are called equally informative if and only if $j \to i$ and $i \to j$.) Only one, the one with the smallest index, of the equally informative items is drawn, and the equally informative items are returned (as tuples) in a list. The plotted Hasse diagram uses as item labels $iL$, a transformation that takes place internally in the packages **sets** or **relations** (cf. Section 3.1). This may look somewhat unesthetic.

Two auxiliary functions for implementing the IITA algorithms are:

```
ob_counter(dataset)
ind_gen(b)
```

The first function computes from a `dataset` for all item pairs the corresponding numbers of observed counterexamples. These values are crucial in the formulations of the IITA algorithms (Section 2.2). This function returns a matrix of the numbers of observed counterexamples for all pairs of items. The second function can be used to generate inductively from a matrix `b` of the numbers of observed counterexamples a set of quasi orders. The inductive generation of the selection set of competing quasi orders is a prime component of the IITA algorithms (Section 2.2). This function returns a list of the inductively generated surmise relations. The main function `iita` (see below) calls `ob_counter` for computation of the numbers of counterexamples, and `ind_gen` for the inductive generation procedure.

Three complex functions of the package **DAKS** realizing the original, corrected, and minimized corrected IITA algorithms (Section 2.2) are, in respective order:

```
orig_iita(dataset, A)
corr_iita(dataset, A)
mini_iita(dataset, A)
```

These functions perform the respective IITA procedures using the `dataset` and the list `A` of prespecified competing quasi orders. The set of competing quasi orders must be passed via the argument `A` manually, so any selection set of surmise relations can be used. The function `iita` (see below) automatically generates a selection set from the data using the inductive generation procedure implemented in `ind_gen` (see above). The latter approach (using `iita`) is common so far, in KST, where the inductive data analysis methods have been utilized for exploratory derivations of quasi orders from the data. The functions `orig_iita`, `corr_iita`, and `mini_iita`, on the other hand, can be used to select among surmise relations for instance obtained from querying experts or from competing psychological theories. All three functions return a vector of the *diff* values corresponding to the competing quasi orders in `A`.

The function that can be used to perform one of the three IITA procedures selectively is:

```
iita(dataset, v)
```

Whereas for the above three functions selection sets of competing quasi orders have to be passed via an argument manually, this function automatically generates a selection set from the `dataset` using the inductive generation procedure implemented in `ind_gen` (see above). The parameter `v` specifies the IITA algorithm to be performed; `v = 1` (minimized corrected), `v = 2` (corrected), and `v = 3` (original). Compared to the above three functions, this function returns, besides the *diff* values corresponding to the inductively generated quasi orders, the derived solution quasi order (with minimum *diff* value) under the selected algorithm and its index in the selection set. (In case of ties in minimum *diff* value, a quasi order with smallest size is returned.)

The package **DAKS** also contains functions which provide the basis for statistical inference methodology (cf. Section 5). The population analog of the previous function that can be used to perform one of the three IITA algorithms in population quantities (in a known population) selectively is:

```
pop_iita(imp, ce, lg, items, dataset = NULL, v)
```

Compared to `iita`, this function implements the three IITA algorithms in population, not sample, quantities; `v = 1` (minimized corrected), `v = 2` (corrected), and `v = 3` (original).

The argument `imp` specifies a surmise relation, and `items` gives the number of items of the domain taken as basis for `imp`. The knowledge structure corresponding to `imp` is equipped with the careless error `ce` and lucky guess `lg` probabilities and the uniform distribution on the knowledge states, and is the known BLIM underlying the population. If `dataset = NULL`, a set of competing quasi orders is constructed based on a population analog of the inductive generation procedure implemented in sample quantities in `ind_gen`. If the `dataset` is specified explicitly, that data are used to generate the set of competing quasi orders based on the sample version of the inductive generation procedure. This function returns the population *diff* values corresponding to the inductively generated quasi orders, all possible response patterns with their population probabilities of occurrence, the population $\gamma_{\sqsubseteq}$ rates corresponding to the inductively generated quasi orders, and the inductively generated selection set (cf. Section 2).

The function for computing population (exact) asymptotic variances of the MLEs *diff* (Section 2.2) is:

```
pop_variance(pop_matrix, imp, error_pop, v)
```

Subject to the selected version to be performed in population quantities, `v = 1` (minimized corrected) and `v = 2` (corrected), this function computes the population asymptotic variance of the MLE *diff*, which here is formulated for the relation and error rate specified in `imp` and `error_pop`, respectively. This population variance is obtained using the delta method (e.g., Casella and Berger 2002), which requires calculating the Jacobian matrix of the *diff* coefficient and the inverse of the expected Fisher information matrix for the multinomial distribution. The cell probabilities of that distribution are specified in `pop_matrix`, a matrix of all possible response patterns and their population occurrence probabilities. Note that the arguments `pop_matrix` and `error_pop` can be obtained from a call to the function `pop_iita` (see above), and that the current version of the package **DAKS** does not support computing population asymptotic variances for the original IITA algorithm. This function returns a single value, the population asymptotic variance of the MLE *diff*.

The function for computing estimated asymptotic variances of the MLEs *diff* is:

```
variance(dataset, imp, v)
```

Subject to the selected version to be performed in sample quantities, `v = 1` (minimized corrected) and `v = 2` (corrected), this function computes a consistent estimator for the population asymptotic variance of the MLE *diff*, which here is formulated for the relation and the data specified in `imp` and `dataset`, respectively. This estimated asymptotic variance is obtained using the delta method (cf. `pop_variance`). In the expression for the exact asymptotic variance (expressed in Jacobian matrix and inverse expected Fisher information), the true parameter vector of the multinomial probabilities is estimated by its MLE of the relative frequencies of the response patterns. Note that the two types of estimators for the population asymptotic variances of the *diff* coefficients obtained using the expected Fisher information matrix and the observed Fisher information matrix yield the same result, in the case of the multinomial distribution. Since computation based on the expected Fisher information matrix is faster, this is implemented in `variance`. Note that the current version of the package **DAKS** does not support computing estimated asymptotic variances for the original IITA algorithm. This function returns the estimated asymptotic variance of the MLE *diff*.

Table 1 summarizes all functions of the package **DAKS**.

Table 1: Summary of the **DAKS** functions

| Function | Short description |
|---|---|
| `corr_iita` | Computing *diff* values for the corrected IITA algorithm |
| `hasse` | Plotting a Hasse diagram |
| `iita` | Computing sample *diff* values and the best fitting quasi order for one of the three IITA algorithms selectively |
| `imp2state` | Transforming from implications to knowledge states |
| `ind_gen` | Inductively generating a selection set |
| `mini_iita` | Computing *diff* values for the minimized corrected IITA algorithm |
| `ob_counter` | Computing numbers of observed counterexamples |
| `orig_iita` | Computing *diff* values for the original IITA algorithm |
| `pattern` | Computing frequencies of response patterns and knowledge states |
| `pop_iita` | Computing population *diff* values and the selection set for one of the three IITA algorithms selectively |
| `pop_variance` | Computing population asymptotic variances |
| `simu` | Data simulation tool |
| `state2imp` | Transforming from knowledge states to implications |
| `variance` | Computing estimated asymptotic variances |

# 4. Demonstrating the package DAKS

## 4.1. An example with real data

We exemplify usage of the package **DAKS** with part of the 2003 Programme for International Student Assessment (PISA; http://www.pisa.oecd.org/) data. The dataset consists of the item responses by 340 German students on a 5-item dichotomously scored mathematical literacy test. This is the `pisa` dataset accompanying the package **DAKS**. This dataset resulted from dichotomizing the original multiple-choice or open format test data. The scores are 1 or 0 for a correct or incorrect response, respectively; there are no missing values in the data. Wordings of the test items used in the assessment are not known (not publicly available).

We first get a general idea of the data.

```
R> head(pisa)
```

```
  a b c d e
1 1 0 0 0 0
2 0 0 0 0 0
3 1 0 0 0 0
4 1 0 0 0 0
5 0 1 0 0 0
6 1 1 0 0 0
```

```
R> pat <- pattern(pisa)
R> pat

$response.patterns

11100 11000 10000 11110 00000
   67    61    41    40    20

$states
NULL

R> sum(pat$response.patterns)

[1] 229
```

We see that the five most frequent response patterns make up for 229 out of the 340 patterns. These are the Guttman patterns of the chain $d \to c \to b \to a$ that can likely be assumed to underlie the data. This is also indicated by the following code.

```
R> apply(pisa, 2, table)

    a   b   c   d   e
0  51  91 167 261 293
1 289 249 173  79  47
```

From items $a$ to $e$, the sample item popularities (proportions-correct) are well-differentiated and strictly decreasing. For instance, item $a$ is most popular (most frequently solved), item $e$ is least popular (least frequently solved). Since we do not know whether the underlying quasi order may or may not be a chain, we next perform IITA analyses of the PISA data.

We start with running the three IITA algorithms on these data. The results are assigned to variables for later analyses.

```
R> mini <- iita(pisa, v = 1)
R> corr <- iita(pisa, v = 2)
R> orig <- iita(pisa, v = 3)
R> orig

$diff
 [1]   82.15616   80.59994   78.63000   74.66784  134.34482  122.66421
 [7]  150.88919  141.51590  133.26889  384.96981  822.49495 1853.68245
[13] 3192.94612

$implications
{(1L, 4L), (1L, 5L), (2L, 4L), (2L, 5L)}

$selection.set.index
[1] 4
```

```
R> corr

$diff
 [1]  143.53305  137.40759  132.21403  115.38470  121.09900  113.42260
 [7]   86.06609   40.93807   33.31853  179.64510  361.25716 1161.38396
[13] 3192.94612

$implications
{(1L, 2L), (1L, 3L), (1L, 4L), (1L, 5L), (2L, 3L), (2L, 4L), (2L, 5L),
 (3L, 4L), (3L, 5L)}

$selection.set.index
[1] 9

R> mini

$diff
 [1]  143.53305  137.39922  132.13348  115.37663  120.16808  110.48656
 [7]   82.54234   38.97623   27.56613  107.39041  242.37313 1079.05432
[13] 2887.72089

$implications
{(1L, 2L), (1L, 3L), (1L, 4L), (1L, 5L), (2L, 3L), (2L, 4L), (2L, 5L),
 (3L, 4L), (3L, 5L)}

$selection.set.index
[1] 9
```

We additionally present the inductively generated selection set of competing quasi orders, because that helps investigating the results obtained from applying the IITA algorithms. (Note that this is practicable when the selection set or the number of items are not too large.)

```
R> ind_gen(ob_counter(pisa))

[[1]]
{(1L, 5L)}

[[2]]
{(1L, 4L), (1L, 5L)}

[[3]]
{(1L, 4L), (1L, 5L), (2L, 5L)}

[[4]]
{(1L, 4L), (1L, 5L), (2L, 4L), (2L, 5L)}

[[5]]
```

```
{(1L, 4L), (1L, 5L), (2L, 4L), (2L, 5L), (3L, 5L)}

[[6]]
{(1L, 3L), (1L, 4L), (1L, 5L), (2L, 4L), (2L, 5L), (3L, 5L)}

[[7]]
{(1L, 3L), (1L, 4L), (1L, 5L), (2L, 4L), (2L, 5L), (3L, 4L), (3L, 5L)}

[[8]]
{(1L, 3L), (1L, 4L), (1L, 5L), (2L, 3L), (2L, 4L), (2L, 5L), (3L, 4L),
 (3L, 5L)}

[[9]]
{(1L, 2L), (1L, 3L), (1L, 4L), (1L, 5L), (2L, 3L), (2L, 4L), (2L, 5L),
 (3L, 4L), (3L, 5L)}

[[10]]
{(1L, 2L), (1L, 3L), (1L, 4L), (1L, 5L), (2L, 3L), (2L, 4L), (2L, 5L),
 (3L, 4L), (3L, 5L), (4L, 5L)}

[[11]]
{(1L, 2L), (1L, 3L), (1L, 4L), (1L, 5L), (2L, 1L), (2L, 3L), (2L, 4L),
 (2L, 5L), (3L, 4L), (3L, 5L), (4L, 5L), (5L, 4L)}

[[12]]
{(1L, 2L), (1L, 3L), (1L, 4L), (1L, 5L), (2L, 1L), (2L, 3L), (2L, 4L),
 (2L, 5L), (3L, 4L), (3L, 5L), (4L, 3L), (4L, 5L), (5L, 3L), (5L, 4L)}

[[13]]
{(1L, 2L), (1L, 3L), (1L, 4L), (1L, 5L), (2L, 1L), (2L, 3L), (2L, 4L),
 (2L, 5L), (3L, 1L), (3L, 2L), (3L, 4L), (3L, 5L), (4L, 1L), (4L, 2L),
 (4L, 3L), (4L, 5L), (5L, 1L), (5L, 2L), (5L, 3L), (5L, 4L)}
```

The quasi order with tenth index in the selection set is a chain. The neighboring quasi orders with indices eight, nine, and eleven are very close to a chain. Therefore we expect the underlying quasi order to be one of these four, most likely. (Note that inspecting the selection set for specific quasi orders can be useful in general, because the selection set only contains a small fraction of all possible quasi orders.)

The corrected and minimized corrected IITA algorithms yield the same solution quasi order, which is close to a chain. The original IITA algorithm selects a quasi order which is clearly different from that returned by the other two algorithms, and which is far from being a chain. This is also reflected by the corresponding *diff* values. They are similar for the corrected and minimized corrected IITA algorithms, and considerably smaller than the *diff* value obtained for the original algorithm. There is evidence that the original IITA algorithm fails in revealing underlying "close-to-chain" quasi orders. Furthermore, fitting the classical Rasch model to this dataset corroborates the chain hierarchy among the five mathematical literacy test items. For details on comparing the different data analysis methods and psychometric approaches,

see Sargin and Ünlü (2009) and Ünlü and Sargin (2008a). The present paper rather is on introducing the R package **DAKS**.

One can use functions of the package **sets**, for example when comparing the solution quasi orders obtained from different IITA algorithms. The symmetric set difference between the solutions of the original and minimized corrected IITA algorithms can be computed by:

```
R> set_symdiff(orig$implications, mini$implications)

{(1L, 2L), (1L, 3L), (2L, 3L), (3L, 4L), (3L, 5L)}
```

The symmetric set difference gives the implications in which the two relations differ. In the example here we see that all implications of the original IITA algorithm solution are contained in the quasi order derived using the minimized corrected IITA algorithm. This is seen by:

```
R> set_is_proper_subset(orig$implications, mini$implications)

[1] TRUE
```

Of course, other functions of the package **sets** can be helpful and used as well.

Graphics are convenient to use and they can present information effectively. The graphic that is used throughout KST is the Hasse diagram. It is utilized for presenting information, not for exploring data. For approaches to graphically exploring KST data based on mosaic plots, see Ünlü and Sargin (2008b). A Hasse diagram can be plotted by:

```
R> hasse(mini$implications, 5)

list()
```

This gives the Hasse diagram of the solution quasi order of the minimized corrected algorithm shown in Figure 1. Note that the returned list of equally informative items is empty; therefore the diagram faithfully presents the quasi order. The plotted Hasse diagram uses as item labels $iL$, a transformation that takes place internally in the packages **sets** or **relations** (cf. Section 3.2).

## 4.2. An example with simulated data

To illustrate the other functions of the package **DAKS**, we start with simulating a dataset. Note that every simulation is individual, in the sense that different results are obtained from simulation to simulation.

```
R> ex_data <- simu(9, 1500, 0.1, 0.1, delta = 0.15)
```

The randomly generated quasi order underlying the simulated data is:

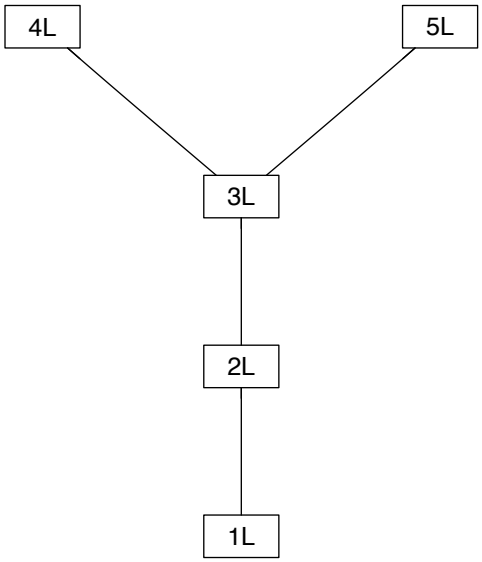



Figure 1: Hasse diagram of the quasi order obtained for the PISA dataset under the minimized corrected IITA algorithm.

```
R> ex_data$implications

{(1L, 6L), (3L, 1L), (3L, 2L), (3L, 6L), (3L, 7L), (5L, 1L), (5L, 2L),
 (5L, 3L), (5L, 6L), (5L, 7L), (7L, 2L), (7L, 6L), (9L, 2L), (9L, 6L),
 (9L, 7L)}
```

In the following, analyses are performed under the corrected IITA algorithm only; under the other two algorithms the analyses are analogous. We run the corrected IITA procedure on the simulated dataset.

```
R> ex_corr <- iita(ex_data$dataset, v = 2)
R> ex_corr

$diff
 [1]  3479.674  3450.734  3355.514  3258.804  3157.678  3148.529  3142.437
 [8]  2680.915  2581.564  2177.801  1742.127  1333.645  1244.936  1203.878
[15]  1006.985  1027.378  1046.564  1446.673  1393.974  1736.810  2297.276
[22]  2725.820  2740.723  3519.965  4149.371  4212.690  4278.585  4361.433
[29]  4470.793  4603.257  5193.763  4645.001  4868.617  6187.025  7571.689
[36]  7287.076  8081.097  8215.758  9854.952 26521.740

$implications
{(1L, 6L), (3L, 1L), (3L, 2L), (3L, 6L), (3L, 7L), (5L, 1L), (5L, 2L),
 (5L, 3L), (5L, 6L), (5L, 7L), (7L, 2L), (7L, 6L), (9L, 2L), (9L, 6L),
 (9L, 7L)}
```

```
$selection.set.index
[1] 12
```

The quasi order obtained by data analysis is the true quasi order underlying the data. (This of course may not always be the case.)

```
R> ex_corr$implications == ex_data$implications

[1] TRUE
```

Next we discuss the functions which provide the basis for statistical inference methodology. The corrected IITA algorithm can be performed in population quantities, yielding information about the population *diff* values, population occurrence probabilities of response patterns, population error rates, and the inductively generated selection set, by:

```
R> pop <- pop_iita(ex_data$implications, 0.1, 0.1, 9,
R+ dataset = ex_data$dataset, v = 2)
R> attributes(pop)

$names
[1] "pop.diff"      "pop.matrix"    "error.pop"     "selection.set"
```

To compare sample with population *diff* values, the sample *diff* coefficient is transformed to become the MLE for the corresponding population *diff* coefficient (see Section 2.2):

```
R> ex_corr$diff / 1500^2

 [1] 0.0015465217 0.0015336597 0.0014913396 0.0014483572 0.0014034125
 [6] 0.0013993461 0.0013966387 0.0011915178 0.0011473617 0.0009679115
[11] 0.0007742786 0.0005927310 0.0005533049 0.0005350570 0.0004475487
[16] 0.0004566123 0.0004651397 0.0006429659 0.0006195440 0.0007719157
[21] 0.0010210118 0.0012114755 0.0012180989 0.0015644287 0.0018441648
[26] 0.0018723067 0.0019015935 0.0019384148 0.0019870190 0.0020458919
[31] 0.0023083390 0.0020644449 0.0021638297 0.0027497888 0.0033651952
[36] 0.0032387004 0.0035915989 0.0036514480 0.0043799785 0.0117874400

R> pop$pop.diff

 [1] 0.0014216299 0.0014110724 0.0013702314 0.0013284010 0.0012920097
 [6] 0.0012777244 0.0012626287 0.0010928491 0.0010563888 0.0008856327
[11] 0.0007148498 0.0005739715 0.0005368436 0.0004987572 0.0004313205
[16] 0.0004421496 0.0004514529 0.0006341078 0.0006279407 0.0007549621
[21] 0.0009297940 0.0011541921 0.0011667470 0.0014536691 0.0016738781
[26] 0.0017241115 0.0017698818 0.0018117440 0.0018501661 0.0018690699
[31] 0.0021182519 0.0018807877 0.0019219018 0.0024877391 0.0029851230
[36] 0.0028642365 0.0031317869 0.0031681007 0.0037913672 0.0099925625
```

The respective sample and population values are quite similar, already for a sample size of 1500. This is obvious given the fact that the sample *diff* values converge in probability (and expectation) to the population *diff* values (Section 2.2).

The quasi order with minimum population *diff* value can be queried:

```
R> pop$selection.set[[which(min(pop$pop.diff) == pop$pop.diff)]]

{(1L, 6L), (3L, 1L), (3L, 2L), (3L, 6L), (3L, 7L), (5L, 1L), (5L, 2L),
 (5L, 3L), (5L, 6L), (5L, 7L), (7L, 2L), (7L, 6L), (9L, 2L), (9L, 6L),
 (9L, 7L)}
```

This quasi order is the true quasi order underlying the simulated dataset. Of course this may not always be the case, especially for smaller sample sizes or higher response error rates. The population analogs are useful for comparing the IITA algorithms (Ünlü and Sargin 2008a).

As mentioned in Section 2.2, the MLEs *diff* are asymptotically normal. Large sample normality with associated standard errors can be used to construct confidence intervals for the population values of and to test hypotheses about the *diff* coefficients. For instance, one could test whether one of two quasi orders has a significantly smaller *diff* value in the population. The quasi orders could, for example, be derived from querying experts. In order to do such a test, the asymptotic variances need to be estimated. Population asymptotic variances and consistent estimators of the latter can be computed using the delta method (cf. Section 3.2).

The estimated asymptotic variance can be computed by:

```
R> var_sample <- variance(ex_data$dataset, ex_data$implications, v = 2)
R> var_sample

 [1] 5.866841e-06

R> sqrt(var_sample)

[1] 0.002422156
```

The corresponding population asymptotic variance is:

```
R> pop_variance <- pop_variance(pop$pop.matrix, pop$selection.set
R+ [[which(min(pop$pop.diff) == pop$pop.diff)]], pop$error.pop
R+ [which(min(pop$pop.diff) == pop$pop.diff)], v = 2)
R> pop_variance

 [1] 4.176084e-06

R> sqrt(pop_variance)

[1] 0.002043547
```

The sample and population values are quite similar. The sample variance is a consistent estimator for the population variance (convergence in probability).

## 5. Conclusion

This paper has introduced the R package **DAKS**. This package contains several basic functions for KST, and it primarily implements the IITA methods for data analysis in KST. Functions for computing various population values and for estimating asymptotic variances are also contained. These tools provide the basis for statistical inference methodology and for further analyses in KST. We have described the functions of the package **DAKS** and demonstrated their usage by real and simulated data examples.

In future research, we plan to implement other fit measures such as the *di* (discrepancy) index (Kambouri, Koppen, Villano, and Falmagne 1994) or the *CA* (correlational agreement) coefficient (van Leeuwe 1974). Functions for computing confidence intervals and for performing hypotheses tests for the *diff* (and other) fit measures will also be implemented. The present functions of the package are to be extended; for example, the function `hasse` should incorporate drawing diagrams for knowledge structures, or the simulation tool could allow for individual response error probabilities for each item.

By contributing the R package **DAKS** we hope to have established a basis for computational work in the so far combinatorial theory of knowledge spaces. Implementing KST procedures in R can help to bring together KST and such other psychometric approaches as item response theory (IRT). A number of R packages are available for IRT; for instance, **eRm**, **ltm**, or **mokken**. KST and IRT are split directions of psychological test theories and are currently compared at a theoretical level (Stefanutti 2006; Stefanutti and Robusto 2009; Ünlü 2007). Using R as an interface between these theories may prove valuable in comparing them at a computational level.

## References

Albert D, Lukas J (eds.) (1999). *Knowledge Spaces: Theories, Empirical Research, and Applications*. Lawrence Erlbaum Associates, Mahwah.

Birkhoff G (1937). "Rings of Sets." *Duke Mathematical Journal*, **3**, 443–454.

Casella G, Berger RL (2002). *Statistical Inference*. Duxbury, Pacific Grove, CA, 2nd edition.

Doignon JP, Falmagne JC (1985). "Spaces for the Assessment of Knowledge." *International Journal of Man-Machine Studies*, **23**, 175–196.

Doignon JP, Falmagne JC (1987). "Knowledge Assessment: A Set Theoretical Framework." In "Beiträge zur Begriffsanalyse: Vorträge der Arbeitstagung Begriffsanalyse, Darmstadt, 1986," pp. 129–140. B.I. Wissenschaftsverlag, Mannheim, Germany.

Doignon JP, Falmagne JC (1999). *Knowledge Spaces*. Springer-Verlag, Berlin.

Falmagne JC (1989). "Probabilistic Knowledge Spaces: A Review." In F Roberts (ed.), "Applications of Combinatorics and Graph Theory to the Biological and Social Sciences," volume 17, pp. 283–303. Springer-Verlag, New York.

Falmagne JC, Koppen M, Villano M, Doignon JP, Johannesen L (1990). "Introduction to Knowledge Spaces: How to Build, Test and Search them." *Psychological Review*, **97**, 201–224.

Kambouri M, Koppen M, Villano M, Falmagne JC (1994). "Knowledge Assessment: Tapping Human Expertise by the QUERY Routine." *International Journal of Human-Computer Studies*, **40**, 119–151.

R Development Core Team (2006). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Sargin A, Ünlü A (2009). "Inductive Item Tree Analysis: Corrections, Improvements, and Comparisons." *Manuscript under revision*.

Schrepp M (2003). "A Method for the Analysis of Hierarchical Dependencies Between Items of a Questionnaire." *Methods of Psychological Research Online*, **19**, 43–79.

Schrepp M (2006). "**ITA 2.0**: A Program for Classical and Inductive Item Tree Analysis." *Journal of Statistical Software*, **16**.

Stefanutti L (2006). "A Logistic Approach to Knowledge Structures." *Journal of Mathematical Psychology*, **50**, 545–561.

Stefanutti L, Robusto E (2009). "Recovering a Probabilistic Knowledge Structure by Constraining its Parameter Space." *Psychometrika*, **74**, 83–96.

Ünlü A (2006). "Estimation of Careless Error and Lucky Guess Probabilities for Dichotomous Test Items: A Psychometric Application of a Biometric Latent Class Model with Random Effects." *Journal of Mathematical Psychology*, **50**, 309–328.

Ünlü A (2007). "Nonparametric Item Response Theory Axioms and Properties Under Nonlinearity and their Exemplification with Knowledge Space Theory." *Journal of Mathematical Psychology*, **51**, 383–400.

Ünlü A, Sargin A (2008a). "Maximum Likelihood Methodology for Diff Fit Measures for Quasi Orders." *Manuscript submitted for publication*.

Ünlü A, Sargin A (2008b). "Mosaics for Visualizing Knowledge Structures." *Manuscript submitted for publication*.

van Leeuwe J (1974). "Item Tree Analysis." *Nederlands Tijdschrift voor de Psychologie*, **29**, 475–484.

**Affiliation:**

Anatol Sargin
Department of Computer Oriented Statistics and Data Analysis
Institute of Mathematics
University of Augsburg
D-86135 Augsburg, Germany
E-mail: anatol.sargin@math.uni-augsburg.de
URL: http://stats.math.uni-augsburg.de/mitarbeiter/sargin/

Ali Ünlü
Department of Computer Oriented Statistics and Data Analysis
Institute of Mathematics
University of Augsburg
D-86135 Augsburg, Germany
E-mail: ali.uenlue@math.uni-augsburg.de
URL: http://www.math.uni-augsburg.de/~uenlueal/

*Journal of Statistical Software* http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume VV, Issue II *Submitted:* yyyy-mm-dd
MMMMMM YYYY *Accepted:* yyyy-mm-dd