



prefmod: An R Package for Modeling Preferences Based on Paired Comparisons, Rankings, or Ratings

Reinhold Hatzinger

WU Wirtschaftsuniversität Wien

Regina Dittrich

WU Wirtschaftsuniversität Wien

Abstract

The first part of this paper describes a series of loglinear preference models based on paired comparisons, a method of measurement whose aim is to order a set of objects according to an attribute of interest by asking subjects to compare pairs of objects. Based on the basic Bradley-Terry specification, two types of models, the loglinear Bradley-Terry model and a pattern approach are presented. Both methods are extended to include subject and object-specific covariates and some further structural effects. In addition, models for derived paired comparisons (based on rankings and ratings) are also included. Latent classes and missing values can be included. The second part of the paper describes the package **prefmod** that implements the above models in R. Illustrational applications are provided in the last part of the paper.

Keywords: paired comparisons, rankings, ratings, preference models, Bradley-Terry, latent classes, missing values, GLMs, R.

1. Introduction

The method of paired comparisons is a well-established technique for measuring relative preferences assigned to certain objects or items of any kind. The aim is to establish an ordering of the objects on a preference scale according to an attribute (we use preference as an umbrella term for orderings in general). The attributes may be based on subjective evaluations of properties of the objects (e.g., tastiness of food, beauty of flowers, perceived risk of portfolios) or on ‘objective’ outcomes under some predefined rules (e.g., strength of football teams, corruptness of countries, quality of scientific journals).

Given a set of objects, the paired comparison (PC) method splits the ordering process into a series of evaluations carried out on two objects at a time. For each pair, the objects are compared and a decision is made which of the two objects possesses more of the attribute. The

comparisons are replicated where the units may be different subjects (usually called judges) or different circumstances under which a comparison is observed.

This paper presents some preference models covering and extending the basic paired comparison formulation by Bradley and Terry and also describes their implementation through the package **prefmod** in R (R Development Core Team 2012), available from the Comprehensive R Archive Network at <http://CRAN.R-project.org/package=prefmod>. Section 2 presents two basic models that are extended in Section 3 by including subject and object-specific covariates and some further structural effects. Alternative response formats (rankings and ratings) are dealt with in Section 4. Section 5 outlines PC models which incorporate missing values and latent classes. In Sections 6 and 7 the **prefmod** package is covered and some illustrational applications are given.

2. The basic paired comparison models

Given J objects (indexed by j and k , $j < k$), the core specification for one paired comparison jk is the Bradley-Terry (BT) model which defines the probabilities to prefer object j to object k ($j \succ k$) or to prefer object k to object j ($k \succ j$) as follows:

$$p(j \succ k | \pi_j, \pi_k) = \frac{\pi_j}{\pi_j + \pi_k} \quad \text{or} \quad p(k \succ j | \pi_j, \pi_k) = \frac{\pi_k}{\pi_j + \pi_k}, \quad \pi_j > 0, \forall j \quad (1)$$

where the π s (also called worth parameters) are the locations of objects on the preference scale.

To ease later presentation, we rewrite (1) as $p(j \succ k | \pi_j, \pi_k) = \sqrt{\pi_j/\pi_k} / (\sqrt{\pi_j/\pi_k} + \sqrt{\pi_k/\pi_j})$, adopting a formulation suggested by Sinclair (1982)

$$p(y_{jk}) = c_{jk} \left(\frac{\sqrt{\pi_j}}{\sqrt{\pi_k}} \right)^{y_{jk}}, \quad (2)$$

where y_{jk} is a response to comparison jk and takes the value of 1, if object j is preferred to k and value of -1 , if object k is preferred to j , c_{jk} is a normalizing constant.

This formulation is used when the recorded paired comparisons are assumed to be independent. All models based on the specification (1) or (2) and relying on the independence assumption are called BT models.

A pattern approach is used in which case the responses of a subject are considered simultaneously. The response is defined as $\mathbf{y} = (y_{12}, y_{13}, \dots, y_{jk}, \dots, y_{J-1,J})$ and the probability for a pattern is given by

$$p(\mathbf{y}) = p(y_{12}, \dots, y_{J-1,J}) = c \prod_{j < k} \left(\frac{\sqrt{\pi_j}}{\sqrt{\pi_k}} \right)^{y_{jk}}. \quad (3)$$

Models based on the joint probability for all PCs are called pattern models.

2.1. The basic loglinear Bradley-Terry model: LLBT

The BT models can be fitted as loglinear models (see, e.g., Fienberg and Larntz 1976; Sinclair 1982; Dittrich, Hatzinger, and Katzenbeisser 1998). It is assumed that the observed number

| Response | Counts | μ_{12} | μ_{13} | μ_{23} | λ_1 | λ_2 | λ_3 |
|---------------|-----------|------------|------------|------------|-------------|-------------|-------------|
| $y_{12} = 1$ | $n_{1>2}$ | 1 | 0 | 0 | 1 | -1 | 0 |
| $y_{12} = -1$ | $n_{2>1}$ | 1 | 0 | 0 | -1 | 1 | 0 |
| $y_{13} = 1$ | $n_{1>3}$ | 0 | 1 | 0 | 1 | 0 | -1 |
| $y_{13} = -1$ | $n_{3>1}$ | 0 | 1 | 0 | -1 | 0 | 1 |
| $y_{23} = 1$ | $n_{2>3}$ | 0 | 0 | 1 | 0 | 1 | -1 |
| $y_{23} = -1$ | $n_{3>2}$ | 0 | 0 | 1 | 0 | -1 | 1 |

Table 1: Design structure for a simple LLBT. The *counts* are the observed numbers of preferences (the dependent variate in the loglinear model), the other columns are covariates for the μ s (dummies representing the comparisons), and covariates for the object parameters λ_1 , λ_2 , and λ_3 .

of preferences $n_{j>k} = n(y_{jk} = 1)$ and $n_{k>j} = n(y_{jk} = -1)$ in a given comparison jk follow a Poisson distribution. Conditional on a fixed number of subjects n_{jk} making the comparison jk , the observed number of preferences follow a binomial distribution.

In the loglinear BT model (LLBT) the expected number $m(y_{jk})$ for the response $y_{jk} \in \{1, -1\}$ is given by

$$m(y_{jk}) = n_{jk} p(y_{jk}).$$

Using equation (2) and standard notation for loglinear models, the linear predictor η for the basic LLBT is given by the equation

$$\eta_{y_{jk}} = \ln m(y_{jk}) = \mu_{jk} + y_{jk}(\lambda_j - \lambda_k), \quad (4)$$

where μ_{jk} is a nuisance parameter for the comparison jk which fixes the marginal distribution to n_{jk} . The λ s represent object parameters and are related to the worth parameters π in (1) or (2) by $\ln \pi = 2\lambda$. In total, we estimate $2\binom{J}{2}$ expected counts.

Table 1 shows the design structure for 3 objects and $\binom{3}{2} = 3$ comparisons.

2.2. The paired comparison pattern model

The pattern models can also be formulated as loglinear models (Dittrich, Hatzinger, and Katzenbeisser 2002). The expected numbers for a whole sequence of preferences \mathbf{y} is given as

$$m(\mathbf{y}) = m(y_{12}, \dots, y_{J-1,J}) = n p(\mathbf{y})$$

where n is the total number of respondents. Given n , the observed numbers of subjects responding with certain patterns $\mathbf{y} = (y_{12}, \dots, y_{J-1,J})$ for $y_{jk} \in \{1, -1\}$ are assumed to follow a multinomial distribution with pattern probabilities $p(\mathbf{y})$ given in equation (3). In total, there are $L = 2\binom{J}{2}$ possible distinct response patterns \mathbf{y} .

The linear predictor $\eta_{\mathbf{y}}$ for the pattern model is

$$\eta_{\mathbf{y}} = \ln m(y_{12}, \dots, y_{J-1,J}) = \mu + \sum_{j<k} y_{jk}(\lambda_j - \lambda_k) = \mu + \sum_{j=1}^J \lambda_j x_j. \quad (5)$$

| Pattern y | | | | μ | λ_1 | λ_2 | λ_3 | $\theta_{12.13}$ | $\theta_{12.23}$ | $\theta_{13.23}$ |
|-------------|----------|----------|--------|-------|-------------|-------------|-------------|------------------|------------------|------------------|
| y_{12} | y_{13} | y_{23} | Counts | Const | x_1 | x_2 | x_3 | $y_{12}y_{13}$ | $y_{12}y_{23}$ | $y_{13}y_{23}$ |
| 1 | 1 | 1 | n_1 | 1 | 2 | 0 | -2 | 1 | 1 | 1 |
| 1 | 1 | -1 | n_2 | 1 | 2 | -2 | 0 | 1 | -1 | -1 |
| 1 | -1 | 1 | n_3 | 1 | 0 | 0 | 0 | -1 | 1 | -1 |
| 1 | -1 | -1 | n_4 | 1 | 0 | -2 | 2 | -1 | -1 | 1 |
| -1 | 1 | 1 | n_5 | 1 | 0 | 2 | -2 | -1 | -1 | 1 |
| -1 | 1 | -1 | n_6 | 1 | 0 | 0 | 0 | -1 | 1 | -1 |
| -1 | -1 | 1 | n_7 | 1 | -2 | 2 | 0 | 1 | -1 | -1 |
| -1 | -1 | -1 | n_8 | 1 | -2 | 0 | 2 | 1 | 1 | 1 |

Table 2: Design structure for a pattern model including dependencies. The *counts* are the observed numbers of preferences (the dependent variate in the loglinear model), the other columns are covariates for the μ (the normalizing constant), variates for the objects and for the dependence parameters.

The μ is constant for all patterns and the covariates x_j for the λ s are generated by summation over all paired comparisons, i.e.,

$$x_j = \sum_{\nu=j+1}^J y_{j\nu} - \sum_{\nu=1}^{j-1} y_{\nu j}.$$

The x_j s for the λ s vary according to the effective pattern \mathbf{y} and are obtained by counting how often object j was preferred minus how often object j was not preferred.

An important feature of the pattern models is the possibility to include dependencies between the decisions, and therefore to abandon the (sometimes unrealistic) assumption of independent decision. Dependencies are defined as associations between pairs of PCs with a common object. It is assumed that the dependencies between responses come from the repeated evaluation of the same objects. Dependence parameters have the form $\theta_{jk.jl}$ and are defined for triples of objects $\{j, k, l\}$, i.e., on pairs of PCs with one common object j . Pairs of PCs are assumed to be independent if there is no common object.

Including dependencies, the probability for a pattern $p(\mathbf{y})$ is extended to

$$p(y_{12}, \dots, y_{J-1,J}) = c^* \prod_{j < k} \left(\frac{\sqrt{\pi_j}}{\sqrt{\pi_k}} \right)^{y_{jk}} \exp(\sum \theta_{jk.jl} y_{jk} y_{jl}).$$

The linear predictor for a certain response pattern then becomes

$$\eta_{\mathbf{y}} = \ln m(\mathbf{y}) = \mu + \sum_{j=1}^J \lambda_j x_j + \sum \theta_{jk.jl} y_{jk} y_{jl}. \quad (6)$$

The design structure for a pattern model for $J = 3$ objects including dependencies is given in Table 2.

In fact, the LLBT and the PC pattern model are generalized linear models (GLMs) and can be fitted using any software capable to estimate GLMs.

3. Extending the basic models

3.1. Undecided responses

In some experiments, the researcher may allow subjects to give an undecided or tie response. Such data also occur naturally in certain sports games such as football. Now a PC response y_{jk} can take three values: $y_{jk} = 1$, if object j is preferred, $y_{jk} = -1$, if object k is preferred, and we additionally define $y_{jk} = 0$ for an undecided response.

Adapting a formulation suggested by Davidson and Beaver (1977) and using (2), the probability for a PC response including ties is given as

$$p(y_{jk}) = c_{jk}^{**} \left(\frac{\sqrt{\pi_j}}{\sqrt{\pi_k}} \right)^{y_{jk}} v^{1-|y_{jk}|}. \quad (7)$$

In this case we define a general undecided effect v for all PCs and the equations defined in (4) are extended by a third equation for an undecided response in the LLBT model

$$\ln m(y_{jk} = 0) = \mu_{jk} + \gamma,$$

where the undecided effect γ is $\ln v$.

This can easily be extended if a different undecided effect is needed for each paired comparison by substituting the common γ with γ_{jk} .

In pattern models, undecided responses can also be incorporated by extending (3). The linear predictor for a pattern model including a general undecided effect then becomes

$$\eta_{\mathbf{y}} = \mu + \sum_{j=1}^J \lambda_j x_j + \gamma u \quad (8)$$

where u is the sum of undecided responses in the corresponding pattern.

To estimate different undecided effects for each paired comparison (8) becomes

$$\eta_{\mathbf{y}} = \mu + \sum_{j=1}^J \lambda_j x_j + \sum_{j < k} \gamma_{jk} u_{jk} \quad (9)$$

where $u_{jk} = 1$ if there is an undecided response in the comparison jk and zero otherwise.

3.2. Categorical and numerical subject effects

Preference decisions can depend on characteristics of the subjects making decisions (Dittrich, Hatzinger, and Katzenbeisser 1998). Inclusion of subject covariates allows the data modeler to move away from the assumption that all subjects have the same preference ordering (given by the object parameters λ or the worth parameters π). Here, we are interested how the object parameters vary according to characteristics of the subjects which can be either categorical (e.g., gender) or numerical (e.g., age).

Categorical subject effects

For the LLBT the starting model equations are given in (4). To illustrate the approach we assume that the subjects are cross-classified according to one or more factors into $s = 1, \dots, S$ groups.

The loglinear representation for the log expected numbers of preferences in comparison jk and for subjects in covariate group s is given by

$$\ln m(y_{jk;s}) = \mu_{jk;s} + y_{jk;s}(\lambda_j + \lambda_{js} - \lambda_k - \lambda_{ks}) \quad \text{for } y_{jk;s} \in \{1, -1\}.$$

where the $\mu_{jk;s}$ are nuisance terms to fix the margins in a multinomial model and are parameters to reproduce the marginal totals for a specific factor level and comparison.

There are different ways to parameterize the model of interest. The approach taken here is to define a reference group, where λ_j and λ_k are the object parameters for that reference group (e.g., $s = 1$) and λ_{js} and λ_{ks} , $s = 2, \dots, S$ are the parameters for the other groups. The main effects λ_s are absorbed into $\mu_{jk;s}$. While the λ_{js} represent the ordering for the reference group, the preference order for the other groups are obtained by adding λ_{js} specific for group s .

Numerical subject effects

When numerical subject characteristics are considered the model development is similar. But now the LLBT and the pattern models have to be extended for each subject.

The LLBT equations for subject i and comparison jk are:

$$\ln m(y_{jk;i}) = \mu_{jk;i} + y_{jk;i}(\lambda_{j;i} - \lambda_{k;i})$$

We model the $\lambda_{j;i}$ through the linear relationship

$$\lambda_{j;i} = \lambda_j + \sum_{p=1}^P \beta_{jp} x_{p;i}$$

where $x_{p;i}$ corresponds to the p th covariate ($p = 1, \dots, P$) for individual i . For each object j there is a separate set of β -parameters which describe the effect of the covariates on that object. λ_j is an intercept for the location of object j for $x = 0$.

All considerations for categorical and numerical subject covariates in the LLBT also apply to pattern models. Effectively, categorical and numerical subject covariates can be incorporated by constructing a separate table (cf. Tables 1 and 2), for each subject group s or each subject i . The basic design matrices are repeated with extra indicators (dummies) for the groups or individuals. This allows us to extend the models to more than one categorical and/or numerical subject covariate in the LLBT and pattern models.

3.3. Object-specific effects

Often objects can be described by certain characteristics which govern the preferences. A group of objects having a certain property might be preferred to objects which do not have this property. Or the preferences could be affected by some numerical properties of the objects.

To incorporate object characteristics (e.g., the price of a product) the following linear reparameterisation is used

$$\lambda_j = \sum_{q=1}^Q \beta_q x_{jq}$$

where x_{jq} is the covariate for characteristic q of object j and β_q is the effect of characteristic q . The LLBT equations for the preferences in comparison jk are given by

$$\ln m(y_{jk}) = \mu_{jk} + y_{jk} \sum_q (x_{jq} - x_{kq}) \beta_q,$$

for pattern models this is

$$\ln m(\mathbf{y}) = \mu + \sum_j x_j \sum_q \beta_q x_{jq}.$$

If the number of object characteristics is small compared to the number of objects, the model will be a simplification of the standard model. Note however, this reparameterisation is useful mainly in situations where the x_{jq} are fixed according to an experimental setting, such as in conjoint studies. For observational x_{jq} , the introduction of an additional error term might be useful (cf., [Turner and Firth 2012](#)).

3.4. Position effect

If the order of the presentation of objects in a comparison is believed to affect the responses we might consider a position effect.

We differentiate between the comparison order by using the notation $jk \cdot j$ if j is presented first and $jk \cdot k$ if k is presented first. When preferring one object to another, there are two distinct expected counts which we denote as $m(y_{jk \cdot j} = 1)$ and $m(y_{jk \cdot k} = 1)$. In both cases j is preferred to k .

The LLBT equations for these two expected counts are

$$\begin{aligned} \ln m(y_{jk \cdot j} = 1) &= \mu_{jk \cdot j} + \lambda_j - \lambda_k + \delta \\ \ln m(y_{jk \cdot k} = 1) &= \mu_{jk \cdot k} + \lambda_j - \lambda_k, \end{aligned}$$

with an extra parameter δ representing a general position effect for the first presented object being chosen.

4. Alternative response formats

4.1. Pattern models for rankings

So far, we have only considered data that originate from (real) paired comparisons. In many studies, however, rankings of objects are collected to get information about the preference order. Subjects are asked directly which object is the most preferred, the second preferred and so on. Such rankings can also be modeled using a pattern approach (model (5) or any extension as presented in the previous sections) by transforming the rankings into (derived) PC patterns ([Francis, Dittrich, and Hatzinger 2010](#)).

Given, for instance, three objects O_1 , O_2 , and O_3 and the preference order $O_1 \succ O_3 \succ O_2$, we can easily derive the according PC response pattern to be $(1, 1, -1)$.

One major difference is the number of possible patterns L . For rankings $L = J!$, whereas for (real) PCs $L = 2^{\binom{J}{2}}$. The reason is that there are many real PC patterns that cannot be generated from a ranking. The patterns left out are *intransitive* patterns, which can not result

| Ranks | | | Derived patterns | | | Counts | μ | λ_1 | λ_2 | λ_3 |
|-------|-------|-------|------------------|----------|----------|--------|-------|-------------|-------------|-------------|
| O_1 | O_2 | O_3 | y_{12} | y_{13} | y_{23} | | Const | x_1 | x_2 | x_3 |
| 1 | 2 | 3 | 1 | 1 | 1 | n_1 | 1 | 2 | 0 | -2 |
| 1 | 3 | 2 | 1 | 1 | -1 | n_2 | 1 | 2 | -2 | 0 |
| 2 | 1 | 3 | -1 | 1 | 1 | n_3 | 1 | 0 | 2 | -2 |
| 2 | 3 | 1 | 1 | -1 | -1 | n_4 | 1 | 0 | -2 | 2 |
| 3 | 1 | 2 | -1 | -1 | 1 | n_5 | 1 | -2 | 2 | 0 |
| 3 | 2 | 1 | -1 | -1 | -1 | n_6 | 1 | -2 | 0 | 2 |

Table 3: Design structure for a pattern model for rankings. The *counts* are the observed numbers of preferences (the dependent variate in the loglinear model), the other columns are covariates for the μ (the normalizing constant) and for the object parameters.

in a ranking. For example, the PC pattern $O_1 \succ O_2, O_2 \succ O_3, O_3 \succ O_1$ with representation $\mathbf{y} = (1, -1, 1)$ which cannot occur with rankings. Another difference to models for (real) PCs is the absence of undecided responses. Moreover, for non-tied rankings not all dependence parameters can be estimated due to their relation to intransitivity (for some discussion see [Dittrich et al. 2002](#)).

Table 3 shows all possible (full) rankings for $J = 3$ objects, derived PC patterns and the design structure for a ranking pattern model.

4.2. Pattern models for ratings

Responses to Likert-type items, often called ratings, are another form of data collection which can be used to obtain preference orderings. Such items (these are the objects in our terminology) are useful for gathering respondents' feelings, opinions, attitudes, etc. It is not uncommon that mean or median values are used to establish an order of these items though based on questionable assumptions. Sometimes, the Likert-type responses are treated as categorical, where frequencies of the high categories for each item are used to determine a ranking of the items. A detailed discussion is given in [Dittrich, Francis, Hatzinger, and Katzenbeisser \(2007\)](#).

Alternatively, we can use the paired comparison pattern model approach to rank Likert items according to the 'measured' dimension (e.g., dangerousness, importance or liking).

The rating pattern models based on Likert-type data can be seen as ranking models naturally including undecided decisions (they also apply to ranking data with ties). To convert Likert-data on sets of items to (derived) PCs we compare pairs of item responses. In the following we assume low rating scores to represent a higher value on the preference scale and that the items all represent the same dimension. Comparing the responses of item j to k , the item j is preferred ($j \succ k$) if j has a lower score and the PC response is $y_{jk} = 1$. The item k is preferred ($k \succ j$) if k has a lower score and $y_{jk} = -1$. Neither of the items is preferred if both items have the same score ($j = k$). This corresponds to an undecided response and $y_{jk} = 0$.

The probabilities for each pair of items jk and a general undecided effect are defined in (7). The linear predictor for this rating pattern model is given by (8). If comparison-specific tie effects are required, the model can be extended as in (9). The set of possible response patterns \mathbf{y} , however, is different.

In contrast to rankings, the undecided responses are inherent in rating pattern models as

items can be rated equally. This is affecting the number of possible derived PC patterns. The number of possible patterns for real PCs including an undecided category is $3^{\binom{J}{2}}$ whereas there are fewer derived patterns for rating data since several rating patterns have the same representation as derived PC patterns (e.g., the rating patterns (1, 1, 2) and (2, 2, 3) are both represented by the unique PC pattern (0, 1, 1)). For details see [Dittrich, Francis, Hatzinger, and Katzenbeisser \(2007\)](#).

The rating pattern models can also be extended to subject and object covariates and, additionally, dependencies can be modeled.

It should be noted, that both ratings and rankings should, in general, not be modeled using LLBTs but rather pattern models. In case of real paired comparisons the LLBT and the independence pattern model coincide. Inherently, the LLBT considers all transitive and intransitive PC patterns. However ranking and rating pattern models are based on fewer response patterns compared to PC pattern models and consequently, the LLBT leads to biased estimates for data without intransitive patterns.

5. Some further extensions to PC pattern models

5.1. Missing values

Missing observations in paired comparison data can occur for several reasons. One is ‘by design’, if deliberately not all possible comparisons are presented to all subjects. Another is that subjects do not know which object to prefer or are unwilling to make a decision, or are tired, etc. In the following we will denote a missing value by NA (not available).

In the LLBT models, missing values can be handled easily, if they occur at random and are not left out systematically by the respondent. The decisions are assumed to be independent (as coming from different subjects) and the n_{jk} can therefore vary for each comparison. But as previously mentioned, the LLBT does not allow for dependencies and gives biased estimates for derived PCs originating from rating and ranking data.

To take incomplete response patterns into account, the composite link approach as suggested by [Thompson and Baker 1981](#) is used ([Dittrich, Francis, Hatzinger, and Katzenbeisser 2012](#)). We distinguish between the *observed* patterns and the *complete* data. Whereas the observed patterns include all possible outcomes including NAs, the complete data are a combination of all possible patterns without NAs (denoted as \mathbf{y}) and all possible nonresponse patterns (denoted by \mathbf{r}). A nonresponse pattern \mathbf{r} describes the NA structure and is of the form $\mathbf{r} = (r_{12}, \dots, r_{J-1,J})$, where r_{jk} is 1 if comparison jk is missing and 0 otherwise. The *observed* and *complete* data are combined by the composite link method.

The set of all possible patterns that can be observed is divided into blocks according to their NA patterns. We use square brackets to denote such blocks. For instance, the block with no missing responses is written as $[\]$, i.e., it consists of all $(2^{\binom{J}{2}})$ in the case of no ties possible patterns \mathbf{y} . In this block, the observed and complete patterns coincide. If, e.g., the comparison jk is missing, the block containing all patterns that possibly could be observed is denoted by $[jk]$. Given three objects, [Table 4](#) shows the data structure for block $[23]$. In total there are $2^{\binom{J}{2}}$ possible blocks.

The missing value models are based on the *complete* data. The joint probability for the \mathbf{y}

| Counts | Observed data | | | Complete data | | | | | |
|------------------|---------------|----------|----------|---------------|----------|----------|-------------|----------|----------|
| | PC patterns | | | PC patterns | | | Nonresponse | | |
| | y_{12} | y_{13} | y_{23} | y_{12} | y_{13} | y_{23} | r_{12} | r_{13} | r_{23} |
| $n_{(1,1,NA)}$ | 1 | 1 | NA | 1 | 1 | 1 | 0 | 0 | 1 |
| $n_{(1,-1,NA)}$ | 1 | -1 | NA | 1 | -1 | 1 | 0 | 0 | 1 |
| $n_{(-1,1,NA)}$ | -1 | 1 | NA | -1 | 1 | 1 | 0 | 0 | 1 |
| $n_{(-1,-1,NA)}$ | -1 | -1 | NA | -1 | -1 | 1 | 0 | 0 | 1 |
| | | | | -1 | -1 | -1 | 0 | 0 | 1 |

Table 4: Data structure for block [23] and PC responses.

and \mathbf{r} patterns is

$$p(\mathbf{y}, \mathbf{r}; \lambda, \psi) = f(\mathbf{y}; \lambda) g(\mathbf{r} | \mathbf{y}; \psi)$$

(1) The first part $f(\mathbf{y}; \lambda)$ is called the outcome model (defined for the complete patterns \mathbf{y}) where λ stands for all parameters depending on the specified outcome model. The linear predictor for the outcome model is $\eta_{\mathbf{y}; \lambda}$ and the expected counts for the response patterns depend on the λ parameters.

(2) The second part $g(\mathbf{r} | \mathbf{y}; \psi)$ is the nonresponse model where the parameters ψ are related to \mathbf{r} and \mathbf{y} depending on the specified nonresponse model. The linear predictor for the nonresponse model is $\eta_{\mathbf{r} | \mathbf{y}}$ and the expected counts for the nonresponse patterns depend on the ψ parameters.

In the following we consider only two of the missing data mechanisms as described by [Rubin 1976](#), i.e., *missing completely at random* (MCAR) and *missing not at random* (MNAR).

The MCAR assumption defined as $g(\mathbf{r}; \psi)$ is valid if the conditional distribution for the nonresponse pattern \mathbf{r} is thought to be independent of the response pattern \mathbf{y} and the parameters ψ are related to \mathbf{r} only.

We consider two different models under the MCAR assumption, where we use α parameters to specify ψ for these models.

In the MCAR model 1, a common parameter α representing the missingness process is assumed which is the same for all comparisons ($\alpha_{jk} = \alpha$). Using the logit formulation for a nonresponse, $\text{logit}(r_{jk}) = \alpha$, and assuming independence, the linear predictor for a nonresponse pattern is given as

$$\eta_{\mathbf{r}; \alpha} = \alpha \sum_{j < k} r_{jk}$$

where the $\sum_{j < k} r_{jk}$ denotes the total number of nonresponses.

The probability for a nonresponse pattern under model 1 is

$$g(\mathbf{r}; \alpha) = \frac{\exp(\eta_{\mathbf{r}})}{\sum_{\ell} \exp(\eta_{\ell})}.$$

For MCAR model 2 we specify an α parameter for each object by reparameterizing α_{jk} to

$\alpha_j + \alpha_k$ and the linear predictor for a nonresponse pattern is

$$\eta_{\mathbf{r};\alpha_j} = \sum_{j=1}^J \alpha_j \left(\sum_{\nu=j+1}^J r_{j\nu} + \sum_{\nu=1}^{j-1} r_{\nu j} \right)$$

The MNAR assumption defined as $g(\mathbf{r} | y_{obs}, y_{mis}; \psi)$ is used if the MCAR assumption does not hold; in other words, the conditional distribution for the nonresponse pattern \mathbf{r} depends on both the observed and the missing values and the parameters ψ are related to \mathbf{r} , y_{obs} , y_{mis} . For the MNAR model we additionally introduce β parameters related to r_{jk} and y_{jk} to characterize the objects.

In this model the linear predictor for a nonresponse pattern is

$$\eta_{\mathbf{r}|\mathbf{y};\alpha_j, \beta_j} = \sum_j \alpha_j \left(\sum_{\nu=j+1}^J r_{j\nu} + \sum_{\nu=1}^{j-1} r_{\nu j} \right) + \sum_j \beta_j \left(\sum_{\nu=j+1}^J r_{j\nu} y_{j\nu} + \sum_{\nu=1}^{j-1} r_{\nu j} y_{\nu j} \right). \quad (10)$$

Here the β parameters depend on the product of the missing indicators and the PC responses and therefore only the missing observations y_{mis} contribute.

To relate the *observed* data with the *complete* data, the composite link approach is applied and the probabilities for some *observed* data are given as follows:

$$\begin{aligned} p(y_{12}, y_{13}, y_{23}; \lambda, \psi) &= f(y_{12}, y_{13}, y_{23}; \lambda) g(0, 0, 0 | y_{12}, y_{13}, y_{23}; \psi) \\ p(y_{12}, y_{13}, \text{NA}; \lambda, \psi) &= \sum_{y_{23}} f(y_{12}, y_{13}, y_{23}; \lambda) g(0, 0, 1 | y_{12}, y_{13}, y_{23}; \psi) \\ &\vdots \end{aligned}$$

The composite links relate each observed pattern to the complete patterns which could generate the observed pattern. The probability for an observed pattern is obtained by summation over all probabilities for the corresponding complete patterns.

For example, the pattern (1, 1, NA) in Table 4, can only be generated by the complete patterns (1, 1, 1) or (1, 1, -1). The observed part $y_{obs} = (1, 1)$ and the missing element y_{mis} is y_{23} , which takes the values 1 or -1. For this example the linear predictor for the observation (1, 1, NA) is

$$\eta_{(1,1,\text{NA});\lambda} = \eta_{(1,1,1)} + \eta_{(1,1,-1)},$$

if no nonresponse model is specified. The specification for the η s are given in (5) if no dependencies are assumed and in (6) when dependencies are included.

If additionally a nonresponse model is specified, e.g., for the MNAR model given in (10), the linear predictor for the observation (1, 1, NA) is

$$\eta_{(1,1,\text{NA});\lambda} = (\eta_{(1,1,1)} + \eta_{r|(1,1,1);\alpha_j, \beta_j}) + (\eta_{(1,1,-1)} + \eta_{r|(1,1,-1);\alpha_j, \beta_j}).$$

To estimate the outcome model $f(\mathbf{y}; \lambda)$ the total likelihood \mathcal{L} is the product of likelihoods for each NA pattern block [-]

$$\mathcal{L} = \mathcal{L}_{[]} \cdot \mathcal{L}_{[12]} \cdots \mathcal{L}_{[12][13]} \cdots \mathcal{L}_{[12] \dots [(J-1)J]}$$

The individual contributions for the block with no missing responses is

$$\mathcal{L}_{[]} = \prod_{\mathbf{y} \in [{}]} \left(\frac{\exp(\eta_{(y_{12}, y_{13}, \dots, y_{J-1}, J)})}{\sum_{\mathbf{y} \in [{}]} \exp(\eta_{\mathbf{y}})} \right)^{n_{\mathbf{y}}}.$$

If, e.g., the comparison 12 is missing, the contribution to the likelihood is

$$\mathcal{L}_{[12]} = \prod_{\mathbf{y} \in [12]} \left(\frac{\exp(\eta_{(1, y_{13}, \dots, y_{J-1, J})}) + \exp(\eta_{(-1, y_{13}, \dots, y_{J-1, J})})}{\sum_{\mathbf{y} \in []} \exp(\eta_{\mathbf{y}})} \right)^{n_{\mathbf{y}}}$$

5.2. Latent subject effects

Variation between responses may be due to different characteristics of the subjects. The models presented in the previous sections allowed for including observed (categorical and numerical) subject covariates. However, there might also be unmeasured or unmeasurable characteristics of the subjects that could affect the responses. Using random effects models is one common approach to account for such heterogeneity (Francis, Dittrich, and Hatzinger 2010).

To introduce random effects in a pattern model (that includes fixed subject effects) we have to specify J random effect components $\delta_{j\ell s}$. For each subject in covariate set s and response pattern ℓ there are J separate random effects one for each object j .

The linear predictor for a standard random effects approach would be

$$\eta_{\ell s} = \sum_{i < k} y_{jk; \ell s} (\lambda_{js} + \delta_{j\ell s} - \lambda_{ks} - \delta_{k\ell s}),$$

where the location of the preference parameter for item j will be shifted up or down for each response pattern ℓ in each subject covariate group s .

The likelihood becomes

$$\mathcal{L} = \prod_{\ell s} \left(\int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} p(y_{\ell s} | \delta_{\ell s}) g(\delta_{\ell s}) d\delta_{1\ell s} d\delta_{2\ell s} \dots d\delta_{J-1; \ell s} \right)^{n_{\ell s}}$$

where $g(\delta_{\ell s})$ is the multivariate probability density function or mixing distribution of the random effects vector.

Francis, Dittrich, and Hatzinger (2010) use an alternative approach, the non-parametric maximum likelihood (NPML) method suggested by Aitkin (1996) and Mallet (1986). The multivariate distribution is replaced by a series of mass point components with unknown probability and unknown location. The mass point approach is a mixture model, where the multinomial (fixed effects) model is replaced by mixture of multinomials.

If the number of components (latent classes) is known, say C , we get C vectors of mass point locations

$$\delta_c = (\delta_{1c}, \delta_{2c}, \dots, \delta_{J-1; c})$$

and the unknown component probability is q_c .

The likelihood becomes

$$\mathcal{L} = \prod_{\ell s} \left(\sum_{c=1}^C q_c p_{\ell sc}(\mathbf{y}_{\ell s} | \delta_c) \right)^{n_{\ell s}}$$

where $\sum_{\ell} p_{\ell sc} = 1$ for each subject group s and mass point c .

To estimate the nonparametric random effects model the EM algorithm is used treating class membership as a missing data problem. A latent class membership indicator $z_{\ell sc}$ for each

ℓ_s combination is introduced and set to one if ℓ_s belongs to class c (zero otherwise). The likelihood now is

$$\mathcal{L} = \prod_{\ell_s} \prod_{c=1}^C (q_c P_{\ell_{sc}}(\mathbf{y}_{\ell_s} | \delta_c))^{n_{\ell_s} z_{\ell_{sc}}}$$

The E-step (re)calculates the expectation $E(z_{\ell_{sc}} | \mathbf{y}_{\ell_s}) = w_{\ell_{sc}}$ which is $P(z_{\ell_{sc}} = 1 | \mathbf{y}_{\ell_s})$. This expected value can be thought of as the posterior probability that a subject characterized by the pattern ℓ_s belongs to class c .

The M-step maximizes the multinomial likelihood w.r.t. the λ s and δ s carried out through a loglinear model with weights $w_{\ell_{sc}}$. The estimate of q_c , the proportion of patterns in latent class c , is obtained from $\hat{q}_c = \sum_{\ell_s} w_{\ell_{sc}} / LC$, where L here is the number of patterns \mathbf{y} .

6. The `prefmod` package

Several existing packages in R (R Development Core Team 2012) will estimate Bradley-Terry models, e.g., **BradleyTerry2** (Turner and Firth 2012), **eba** (Wickelmaier and Schmid 2004), and **psychotree** (Strobl, Wickelmaier, and Zeileis 2011). These packages are limited to true PC data and cannot deal with dependencies. The **prefmod** package (Hatzinger 2012) provides an integrated suite of functions which can cover the wide variety of models suitable for PC and derived PC data as discussed above. Apart from model fitting routines there are several example data sets, functions to simulate PC data and to transform raw data.

The two basic model specifications are the LLBT and the pattern model (as presented in Section 2). The names of the functions that relate to these two models echo these names, starting with either `llbt` or `patt`.

6.1. General usage

The work flow to estimate a PC model is: (1) prepare the data, (2) set up a design matrix, (3) fit the model, and (4) display the results. Since the loglinear PC models are GLMs, the model fitting can be accomplished using the R function `glm` or preferably (because it is more efficient), the function `gnm` from the **gnm** package (Turner and Firth 2011). The tricky part here is to set up the design matrices. This is supported by the functions `llbt.design` and `patt.design`. Once having devised the design, model fitting using one of the generalized linear model functions is straightforward. Each of the four steps is explained in more detail below:

Data preparation

The result must be a data frame on an individual level (with one line per subject), where the leftmost columns must be the responses optionally followed by columns for subject covariates. Three main types of responses are covered by the `prefmod` package: (real) paired comparisons, rankings and ratings.

Paired comparisons: Responses have to be coded as consecutive integers, e.g., (0, 1), (1, 2), or (1, 2, 3), if ties are present. The smallest value corresponds to (highest) preference for the first object in a comparison. The codings (1, -1), (2, 1, -1, -2), (1, 0, -1), (2, 1, 0, -1, -2) etc. can also be used. Then, negative numbers denote not preferred and 0 undecided. (In general, responses may be ordinal, e.g., (-2, -1, 0, 1, 2), but they are internally reduced to

two or three categories, i.e., -1 and -2 are treated the same.) The columns for the PC responses correspond to the comparisons where the mandatory order of comparisons is (12), (13), (23), (14), (24), (34), (15), (25), etc.

Rankings and ratings: Each column corresponds to an object, the responses have to be coded in a similar way to PC responses.

Subject covariates: If present, they have to be specified as numeric vectors (not as factors). Levels of categorical covariates have to be represented by consecutive integers starting with 1.

Missing values are represented by `NA` and are allowed only for responses but not for subject covariates.

Design matrix

The two functions `llbt.design` and `patt.design` generate data frames containing the design structures from the above mentioned data for LLBT and pattern models, respectively. Both functions allow for specifying categorical and numeric subject covariates (using the arguments `cat.scovs` and `num.scovs`). For categorical subject covariates, a separate design matrix is generated for every combination of covariate levels and these matrices are stacked to form the overall design matrix. For numeric subject covariates design matrices for every individual are generated and again stacked to form the overall design matrix. The option `casewise` (which is set to `TRUE` by default when numeric subject covariates are present) applies in two situations. If the number of combined levels of the factors specified in `cat.scovs` exceeds the number of subjects, a smaller overall table is generated. Secondly, the option `casewise` allows for fitting a suitably dimensioned null model (without subject covariates) to serve as a reference model.

Common to both functions `llbt.design` and `patt.design` there is also the option to specify a data frame for object-specific covariates (`objcov`). For both subject and object-specific covariates the corresponding vectors are generated in the resulting design structure.

For pattern models there are a few more arguments. The `resptype` argument allows the user to set the response format to `paircomp` (for real PCs), `rating` or `ranking`. Covariates for dependence parameters are generated through the `ia` argument. The preference coding of the responses can be reversed using `reverse = TRUE`. Then the highest value for codings with nonnegative integers denotes a preference for the first object or if the coding is of the form $(-1, 1)$, it will be interpreted as $(1, -1)$, i.e., 1 denotes the preference for the second object.

It is worth noting that the function `patt.design` automatically provides dummies for undecided response categories for each comparison (for rating models and for PC models, if ties are present). The function `llbt.design` produces dummies for each response category. They are labelled `g0` and `g1` (in case of no undecided response) or `g0`, `g1`, and `g2` (in presence of an undecided response category). The dummy `g1` can then be used to fit a global undecided effect, the interaction term `g1:mu` allows for estimating comparison-specific undecided effects. Moreover, the function `llbt.design` generates a factor for the comparisons (`mu`). Both functions create a factor `CASE` with a separate level for each row of the original data matrix if the argument `casewise` was set to `TRUE`. For both functions, the output is a data frame of either class `llbtDES` or `pattDES` containing all variates to fit the corresponding models.

Model fit

Both type of models can be fitted using `glm` or `gnm` using `family = poisson` and the above design as the `data` argument. If categorical subject covariates are present, the highest interaction term amongst them must be present in the linear predictor as a nuisance term, having the sole purpose of fixing the marginal distribution of the contingency table. For fitting numerical subject (and optionally additional categorical) covariates this interaction term is instead represented by the factor `CASE`. For LLBT models this interaction must include the comparison factor `mu`, i.e., `mu` must always be present. It is preferable to use `gnm` rather than `glm`, since these nuisance terms can be specified in `gnm`'s `eliminate` argument resulting in faster computations (see [Hatzinger and Francis 2004](#)), fewer memory requirements, and the suppression of the nuisance parameter estimates from the display of the results.

Effects of the subject covariates are specified using interaction terms between the subject and the object(-specific) covariates.

Display results

Numerical summaries are obtained from the generalized linear model functions as usual. The two functions `llbt.worth` and `patt.worth` applied to the output of `glm` or `gnm` calculate the worth parameters for all objects (or combination of objects if reparameterized using object-specific covariates) optionally grouped for all combinations of categorical subject covariates. The output is a matrix of worth parameters with columns for all combinations of levels of the categorical subject covariates which can then be plotted using `plot`. Optionally, the original estimates (prior to conversion to the worths) can be calculated and plotted. The functions do not directly work for models including numerical subject covariates. However, the resulting estimates can be used for generating appropriate plots (for more details see `?llbt.worth` or `?patt.worth`).

6.2. Alternative work flow

To ease computations some convenience functions have been devised that directly perform model fitting without the need to set up a design matrix. These are `llbtPC.fit` for LLBT models and `pattPC.fit`, `pattR.fit`, and `pattL.fit` for pattern models. The capital letters denote: PC for (real) paired comparisons, R for rankings, and L for ratings/Likert type responses. They all require the data prepared as described in the previous section.

There are some limitations when using these convenience functions. These concern mainly the lack of support for object-specific and continuous subject covariates. Only a general undecided effect can be fitted. Moreover, there is no general formula interface and hence no special model terms (e.g., interaction terms with single objects) can be specified. However, the fitting is based on the likelihood that is directly computed from the internally constructed design matrices. This allows for a higher number of subject covariates and the treatment of missing values in pattern models using a composite link approach.

By default, all objects are included in the models (without any requirement to explicitly specify them). Categorical subject effects can be included using the `formel` and `elim` arguments. `formel` specifies the actual model to be fitted. For instance, if specified as `formel = ~ SEX` a different preference scale for the objects will be estimated for males and females. For two or more covariates, the operators `+` or `*` can be used to model main or interaction effects,

| Response | Model | Data | Design matrix | Estimation | Notes |
|----------|---------|------|--------------------------|-------------------------|----------|
| Real PCs | LLBT | Data | <code>llbt.design</code> | <code>glm, gnm</code> | 1–3, 5–7 |
| | | Data | → | <code>llbtPC.fit</code> | 1, 5, 6 |
| | Pattern | Data | <code>patt.design</code> | <code>glm, gnm</code> | 1–5, 7 |
| | | Data | → | <code>pattPC.fit</code> | 1, 4–6 |
| Ranking | Pattern | Data | <code>patt.design</code> | <code>glm, gnm</code> | 1–3, 7 |
| | | Data | → | <code>pattR.fit</code> | 1, 6 |
| Rating | Pattern | Data | <code>patt.design</code> | <code>glm, gnm</code> | 1–5, 7 |
| | | Data | → | <code>pattL.fit</code> | 1, 4–6 |

Table 5: Overview of basic functions. Notes: The functions support (1) categorical subject covariates, (2) continuous subject covariates, (3) object-specific covariates, (4) dependencies, (5) tied responses, (6) missing responses, (7) R standard output, i.e., standard extractor functions such as `vcov` or `AIC` usable.

respectively. The operator `:` is not allowed. The specification for `elim` follows the same rules as for `formel`. However, `elim` specifies the basic contingency table to be set up but does not specify any covariates to be fitted. This feature allows for the successive fitting of nested models to enable the use of deviance differences for model selection. It is essential, when comparing models through deviance differences that the `elim` formula is the same.

Where applicable, parameters for undecided responses (argument `undec`) and dependencies (argument `ia`) can be included into the models.

The functions for calculating worth parameters and for plotting the preference scales described earlier are also defined for the output of the convenience functions described here and can be used analogously.

Table 5 gives an overview of the work flow and the basic functions of the `prefmod` package.

6.3. Extended functionality

Missing responses

In the LLBT models, missing responses under the missing completely at random (MCAR) assumption do not pose any problem since the number of respondents can vary across comparisons. This is not so straightforward for pattern models. Here the dependent variate is the number of subjects showing a particular (complete) response pattern. If certain comparisons are missing, standard GLMs do not apply. The convenience functions `pattPC.fit`, `pattR.fit`, and `pattL.fit` have been implemented to allow for including response patterns where some comparisons are missing (MCAR). This enables the modeling of specialized response designs such as partial rankings (only a subset of objects is ranked) or “best-worst” choices. Based on a composite link approach the total likelihood of these models is maximized from the sum of individual likelihoods that are calculated for each block of the design structure with differing response/nonresponse patterns (for details see Section 5).

The convenience functions provide some treatment of missing not at random (MNAR) mech-

anisms. Setting the argument `NItest` to `TRUE` gives separate estimates for the object parameters for the group of respondents showing complete responses and the group having at least one `NA` in the responses. If the estimates differ significantly there is some evidence for different response behaviour in the groups.

For real paired comparisons, additionally two different nonresponse models can be specified that allow for the evaluation of nonignorable missings (MNAR). When the outcome model is specified as above, the nonresponse models either assume a common probability for a nonreponse for all objects (argument `MIScommon`) or different nonresponse probabilities for the objects (arguments `MISalpha` and `MISbeta`). `MISalpha` and `MISbeta` are specified using logical vectors with entries for each object. A `TRUE` specifies that the corresponding parameter for the object should be estimated. `MISalpha` defines the set of `NA` indicator parameters and `MISbeta` defines the set of interactions between the `NA` indicator parameters and the object parameters of the outcome model. If only `MISalpha` is specified, the result is an MCAR model. The utility function `checkMIS` facilitates the specification of `MISalpha` and `MISbeta`. Applied to the data, `checkMIS` returns a logical vector that determines for each object whether missing values have been recorded at all. If not used in the context of model fitting, `checkMIS` returns some descriptive statistics on the response structure in the data.

Latent class preference models

The function `pattnpml.fit` allows for the computation of latent class models for real and derived paired comparisons utilizing a nonparametric maximum likelihood (NPML) estimation method (Aitkin 1996). The function is a wrapper for `alldistPC` which in turn is a modified version of the function `alldist` from the `npmlreg` package (Einbeck, Darnell, and Hinde 2012). This function extends the NPML approach to allow fitting of overdispersed paired comparison models. It assumes that overdispersion arises because of dependence in the patterns. Fitting a non-parametric random effects term is equivalent to specifying distinct latent classes of response patterns.

Prior to using `pattnpml.fit` a design matrix has to be generated with `patt.design` and supplied using the `design` argument. Model specification is accomplished with two model formula arguments, `formula` for the fixed effects and `random` for the random effects. Fitting latent class models can be achieved by specifying the paired comparison objects as additive terms in the random part of the model formula. A separate estimate for each item and for each mass point component is produced. The number of components of the finite mixture has to be specified beforehand using the argument `k`.

Fitting subject covariate models with the same effect for each mass point component is achieved by the following specifications: the `formula` part must contain a) the interaction of all subject factors (this corresponds to the specification of the `eliminate` argument when using `gmm` for pattern models) and b) an interaction of the chosen subject covariates with the objects. The `random` argument contains only the objects and gives a model with a set of random intercepts (one set for each mass point).

Fitting subject covariate models with a different effect for each mass point component (sometimes called random coefficient models, see Aitkin, Francis, Hinde, and Darnell 2009, pp. 497) is possible by additionally specifying an interaction of the subject covariates with the objects in the `random` part. Thus the setting `random = ~ x:(o1 + o2 + o3)` gives a model with a set of random slopes (one set for each mass point) and a set of random intercepts, (again one

set for each mass point).

The functions `patt.worth` and `plot` can again be used for displaying the results.

Models for multivariate responses

Currently, the convenience functions `pattLrep.fit` and `pattRrep.fit` are implemented. They allow the fitting of models for ratings and rankings, respectively. The term ‘multivariate’ applies to two scenarios: Each subject evaluates the objects repeatedly over time or each subject evaluates the objects according to different attributes, e.g., different politicians are ranked according to their social and their economic competence (cf., [Dittrich, Francis, Hatzinger, and Katzenbeisser 2006](#)). The function `pattRrep.fit` supports the specification of within-comparison dependencies, `pattLrep.fit` additionally between-comparison dependencies.

6.4. Some utilities

Aggregated data

Sometimes, data are available as counts. Since all **prefmod** functions for setting up designs and fitting models expect the raw data as individual response patterns, count vectors have to be transformed into individual-level data. The function `expand.mat` expands a matrix of unique response patterns into individual response patterns by repeating each pattern by the corresponding count. The unique response patterns may also include values for subject covariates.

Simulating data

Random data for all response types can be generated using the functions `simPC`, `simR`, and `simL` (the capital letters again denote paired comparisons, rankings, and ratings/Likert type responses). The preference order for the objects can be specified using a vector of predefined worth values using the argument `worth`. If they do not sum up to one, they are normalized.

7. Examples

This section demonstrates the use of (most) **prefmod** functions and roughly follows the development of Sections 2–5. All analyses are based on real data. However, the steps have been simplified to ease presentation. Thus, the analyses are only illustrational. All datasets are contained in **prefmod**.

7.1. Training delivery modes: LLBT with different types of covariates

The dataset `trdel` contains data from a paired comparison study to investigate which of five training delivery modes trainees prefer ([Schöll and Veith 2011](#)). The modes were computer-based (CO), TV-based (TV), paper-based (PA), audio-based (AU) and classroom-based (CL) training. Study participants were unemployed persons in the labour market training of the Austrian labour market service (AMS). To account for trainee characteristics that might affect the preference order the variables gender, age, and learning personality type were recorded.

These variables were coded as `sex` (1 male, 2 female), `age` (numeric in years), `ltype` (1 accomodator, 2 diverger, 3 converger, 4 assimilator). The learning types were identified from a questionnaire (Kolb and Kolb 2005). The theory of Kolb and Kolb portrays “two dialectically related modes of grasping experience – Concrete Experience (CE) and Abstract Conceptualization (AC) – and two dialectically related modes of transforming experience – Reflective Observation (RO) and Active Experimentation (AE).” The learning types are characterized as accomodating (CE and AE), diverging (CE and RO), assimilating (AC and RO), and converging (AC and AE). The `trdel` dataset is an excerpt from the original data. Missing values and a few more subject variables have been removed. A more detailed analysis is given in Schöll and Veith (2011).

For illustrational purposes we first fit two models, each including just one of the subject covariates `ltype` or `age`. We evaluate the effect of the categorical variable `ltype` by using the convenience function `llbtPC.fit`. All objects contained in the data frame `trdel` are included by default. We get

```
R> data("trdel")
R> objnames <- c("CO", "TV", "PA", "AU", "CL")
R> mod.tr.ltype <- llbtPC.fit(trdel, nitems = 5, obj.names = objnames,
+   formel = ~ ltype, elim = ~ ltype)
R> mod.tr.ltype
```

Call:

```
gnm(formula = formula, eliminate = elim, family = poisson, data = dfr)
```

Coefficients of interest:

| | CO | TV | PA | AU | CL | CO:ltype2 | CO:ltype3 |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| | 0.073124 | -0.273088 | -0.220487 | -0.557584 | NA | 0.011227 | -0.239959 |
| CO:ltype4 | | | | | | | |
| | -0.172617 | -0.030283 | -0.502874 | -0.304142 | -0.012386 | -0.129191 | 0.001032 |
| AU:ltype2 | | | | | | | |
| | -0.009732 | -0.381967 | -0.165364 | NA | NA | NA | |

Deviance: 33.98696

Pearson chi-squared: 33.35206

Residual df: 24

The worth estimates and their graphical representation are obtained from

```
R> w <- llbt.worth(mod.tr.ltype)
R> plot(w, log = "y")
```

and displayed in Figure 1 (the argument `log = "y"` allows to plot the estimates on a log scale).

For fitting models with numerical subject covariates we cannot use the convenience function `llbtPC.fit`. We use `gnm` instead and have to generate a design structure first (we display the first two lines).

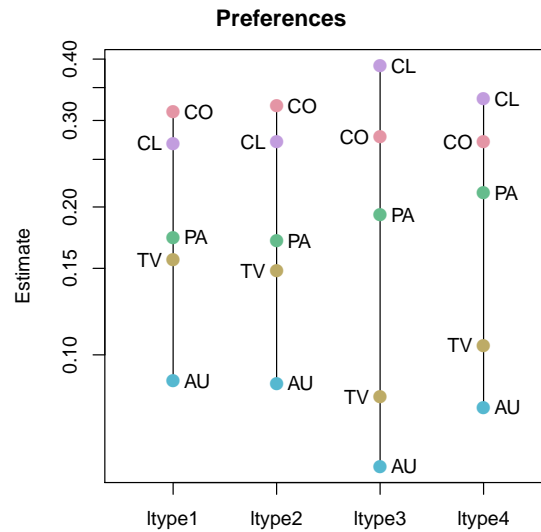


Figure 1: Worth estimates for presentation modes.

```
R> dsg <- llbt.design(trdel, 5, objnames = objnames, num.scovs = "age")
R> head(dsg, n = 2)
```

```
  y mu g0 g1 CO TV PA AU CL age CASE
1 0 1 1 0 1 -1 0 0 0 52 1
2 1 1 0 1 -1 1 0 0 0 52 1
```

The factor `mu` describes the comparisons (in Table 1 `mu` is represented by dummy variables), `g1` and `g2` are dummies for the response categories (see Section 6.1) and `CASE` is a factor representing the subjects.

The `eliminate` argument of `gnm` has to be specified as `mu:CASE` if there is any continuous subject covariate in the model. The `mu:CASE` interaction determines which comparison has been made by which subject.

```
R> mod.tr.age <- gnm(y ~ (CO + TV + PA + AU + CL) * age,
+   family = poisson, eliminate = mu:CASE, data = dsg)
R> mod.tr.age
```

Call:

```
gnm(formula = y ~ (CO + TV + PA + AU + CL) * age, eliminate = mu:CASE,
     family = poisson, data = dsg)
```

Coefficients of interest:

| CO | TV | PA | AU | CL | age | CO:age |
|---------|---------|---------|---------|----|-----|----------|
| 0.64041 | 0.16716 | 0.15176 | 0.08330 | NA | NA | -0.02584 |
| TV:age | PA:age | AU:age | CL:age | | | |

```
-0.02532 -0.01574 -0.03024      NA
```

```
Deviance:          2408.512
Pearson chi-squared: 2007.251
Residual df:       1972
```

For a graphical display of the worth parameters we have to evaluate the fitted linear predictor for a range of values of age, and transform them to worths.

The main effect estimates for the objects (CO through CL) are the intercepts, the interactions of the objects with age (CO:age through CL:age) are the slopes, e.g., $est = CO + (CO:age) * x$, where we use $x = 15, \dots, 55$, the range of observed age values. The parameter estimates relating to the reference object have to be set to zero. The `gnm` function `parameters` is used to obtain the coefficients with NAs replaced by zeros.

```
R> est <- parameters(mod.tr.age)
R> estmat <- do.call("cbind", lapply(15:55,
+   function(x) est[1:5] + est[7:11] * x))
```

The matrix `estmat` now consists of the predicted values for the linear predictor. The rows represent the objects and the columns contain the estimated linear predictor for each age value. To obtain the worth we now have to normalize the transformed estimates using

```
R> wmat <- apply(estmat, 2, function(obj) exp(2 * obj)/sum(exp(2 * obj)))
```

This constrains the worths to sum to one for each age value. The worth plot for age (see Figure 2) is obtained by

```
R> col <- rainbow_hcl(5)
R> plot(c(10, 60), range(wmat), type = "n")
R> for (i in 1:5) lines(15:55, wmat[i, ], col = col[i])
R> text(rep(10, 5), wmat[, 1], objnames)
```

We can see an increasing preference for class-room based training with age whereas the preference for computer-based learning materials decreases.

For a more complex analysis we include all subject covariates into the model. Additionally, we define an object-specific covariate type of presentation media (`pres`), where CO, TV, and AU are coded with 0, and PA and CL with 1. We redefine the design structure as

```
R> PRES <- data.frame(pres = c(0, 0, 1, 0, 1))
R> dsg <- llbt.design(trdel, 5, objnames = objnames, objcov = PRES,
+   cat.scovs = c("sex", "ltype"), num.scovs = "age")
```

The maximal model includes the highest interaction term between `sex`, `ltype`, and `age`. Since `age` is continuous, the `eliminate` argument is specified using the `mu:CASE` interaction.

```
R> mod.tr.max <- gnm(y ~ (CO + TV + PA + AU + CL) * (sex * ltype * age),
+   family = poisson, eliminate = mu:CASE, data = dsg)
```

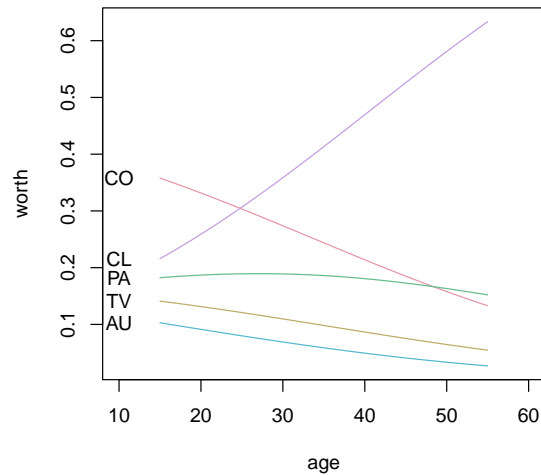


Figure 2: Estimated worth for presentation types by age.

To evaluate if the reduction of the objects to type of presentation media is admissable we fit

```
R> mod.tr.max.pres <- gnm(y ~ pres * (sex * ltype * age),
+   family = poisson, eliminate = mu:CASE, data = dsg)
```

A command such as `anova(mod.tr.max, mod.tr.max.pres)` would show the deviance difference of 281.92 on $df = 48$ indicating that the objects should not be reparameterized.

We continue model selection dropping non significant terms from the maximal model such as the three-way interaction. After some model simplification we end up with

```
R> mod.tr.min <- gnm(y ~ (CO + TV + PA + AU + CL) * (sex + ltype * age),
+   family = poisson, eliminate = mu:CASE, data = dsg)
```

Constructing worth plots follows the same lines as before but now complicated by the presence of the main effect `sex` and the `age:ltype` interaction. A separate set of fitted linear predictors for the objects has to be computed for each combination of levels of `ltype` and `sex` (e.g., `CO + CO:sex2 + CO:ltype3 + (CO:age + CO:ltype3:age) * x`).

7.2. Agresti's baseball data: Aggregated data, position effect

The data are on the results of the 1987 season for professional baseball teams in the Eastern Division of the American League (see [Agresti 2002](#), p. 437). The teams were Milwaukee (MIL), Detroit (DET), Toronto (TOR), New York (NY), Boston (BOS), Cleveland (CLE) and Baltimore (BAL). The question is if there was a home team advantage. The data

```
R> data("baseball")
R> baseball
```

```
[1] 4 3 4 2 4 2 4 3 4 3 2 4 6 1 6 0 4 3 4 3 4 2 6 1 4 2 4 2 5 2 6 0 4
[34] 3 6 0 6 1 6 0 2 4 3 3 5 2 3 4 3 3 1 5 5 2 1 5 5 2 3 3 2 4 5 2 3 3
[67] 4 3 3 4 2 4 5 2 5 1 6 1 4 2 6 1 4 3
```

is given in standard order, i.e., the first two values describe games of Milwaukee vs. Detroit, Milwaukee won 4 and Detroit 3. The data vector has 84 values. For values 1 through 42, the first team in a comparison was the home team, for values 43 through 84, the first team in a comparison was the away team. Thus, the first two values refer to Milwaukee vs Detroit when Milwaukee was the home team.

Since the data is in aggregated form we cannot directly use `prefmod`'s design generating functions. However, a little trick helps.

We can use `llbt.design` to set up the basic design structure, and then modify the structure, adding in the aggregated counts. We form a simple invented data set (`pseudo`) with 21 columns (the number of comparisons for 7 objects) and two lines (one for matches played at home and one for matches played away) augmented with an extra column for a home/away covariate `cov`. The actual values of the observed comparisons stored in the 21 columns do not matter at this stage as the y variate (counts) generated here will later be replaced by the actual number of wins and losses. So we make all values in the first line 1 and all values in the second line -1 .

```
R> pseudo <- data.frame(rbind(c(rep(1, 21), cov = 1), c(rep(-1, 21),
+   cov = 2)))
```

We can now use `llbt.design` to generate the basic design structure. The above specification generates two tables, where the $-1/1$ responses are converted to 0/1 counts of the first team winning.

```
R> objnames <- c("MIL", "DET", "TOR", "NY", "BOS", "CLE", "BAL")
R> dsg <- llbt.design(pseudo, nitems = 7, objnames = objnames,
+   cat.scovs = "cov")
R> head(dsg, n = 2)
```

```
  y mu g0 g1 MIL DET TOR NY BOS CLE BAL cov
1 1  1  1  1  0  1 -1  0  0  0  0  0  1
2 0  1  0  1 -1  1  0  0  0  0  0  0  1
```

```
R> tail(dsg, n = 2)
```

```
  y mu g0 g1 MIL DET TOR NY BOS CLE BAL cov
83 0 21  1  0  0  0  0  0  0  1 -1  2
84 1 21  0  1  0  0  0  0  0 -1  1  2
```

All we have to do now is to set up a dummy for playing at home/away (`home`) and plug in the factor for the comparisons (now 42 levels) and the real data into `dsg`. The new `home` variable is a dummy variable for whether or not the winner is at home (whereas `cov` indicated whether the first team is at home or away).

```
R> home <- c(rep(1:0, 21), rep(0:1, 21))
R> dsg$mu <- gl(42, 2)
R> dsg$y <- baseball
```

We fit two models, with and without considering a possible home advantage.

```
R> mod.bb.0 <- gnm(y ~ MIL + DET + TOR + NY + BOS + CLE + BAL,
+   eliminate = mu, family = poisson, data = dsg)
R> mod.bb.1 <- update(mod.bb.0, . ~ . + home)
R> anova(mod.bb.0, mod.bb.1)
```

Analysis of Deviance Table

```
Model 1: y ~ MIL + DET + TOR + NY + BOS + CLE + BAL - 1
Model 2: y ~ MIL + DET + TOR + NY + BOS + CLE + BAL + home - 1
```

| | Resid. Df | Resid. Dev | Df | Deviance |
|---|-----------|------------|----|----------|
| 1 | 36 | 44.1 | | |
| 2 | 35 | 38.6 | 1 | 5.41 |

The model including a home advantage effect provides a reasonable fit (residual deviance is 38.64 on $df = 35$). The results indicate a strong home advantage effect, with the estimated parameter for home being 0.302. The home team of two evenly matched teams therefore has a winning probability of

```
R> plogis(coef(mod.bb.1)["home"])

home
0.575
```

7.3. Critchlow and Fligner's salad data: Rankings, object-specific covariates

Critchlow and Fligner (1991) report data on an experiment, where 32 subjects rank four salad dressings according to tartness with a rank of 1 being assigned to the dressing judged as being the most tart. The salads A to D had varying concentrations of acetic and gluconic acid. The four pairs of concentrations (percentages) $A = (0.5, 0)$, $B = (0.5, 10)$, $C = (1, 0)$, and $D = (0, 10)$. The question was whether the data could be described more parsimoniously using the acid covariates.

For fitting models with object-specific covariates we need a design matrix.

```
R> data("salad")
R> conc <- data.frame(acet = c(0.5, 0.5, 1, 0), gluc = c(0, 10, 0, 10))
R> sal <- patt.design(salad, nitems = 4, resptype = "ranking",
+   objcov = conc)
R> m.sal.0 <- gnm(y ~ A + B + C + D, family = poisson, data = sal)
R> m.sal.1 <- gnm(y ~ acet + gluc, family = poisson, data = sal)
R> anova(m.sal.1, m.sal.0)
```


Analysis of Deviance Table

```

Model 1: y ~ acet + gluc
Model 2: y ~ A + B + C + D
  Resid. Df Resid. Dev Df Deviance
1         21         22.8
2         20         22.2  1    0.499

```

The reparameterized model obviously provides a satisfactory fit. Using `patt.worth` to calculate the worth parameters additionally gives a table of the object-specific covariate values for each object.

```

R> patt.worth(m.sal.1)

      worth
A 0.08743
B 0.49088
C 0.25001
D 0.17167
attr(,"objtable")
  acet gluc objects in groups
1  0.5   0                A
2  1.0   0                C
3  0.0  10                D
4  0.5  10                B

```

7.4. Music style preference data: Ratings, one subject covariate

The US General Social Survey 1993 contains ratings (five point Likert-type response scale) on various musical styles (1 ... like very much – 5 ... dislike very much). To illustrate the usage of `pattL.fit` we chose four styles: classical, country, rap, and contemporary rock music, and the variable age categorized into three groups: 1 ... –29, 2 ... 30–49, and 3 ... 50+.

```

R> data("music")
R> dat <- music[, c("clas", "coun", "rap", "conr", "age")]
R> age3 <- as.numeric(cut(dat$age, breaks = c(0, 29, 49, 99)))
R> dat$age3 <- age3
R> mod.mus <- pattL.fit(dat, 4, formel = ~ age3, undec = TRUE)

```

The results of fitting this model are

```
R> mod.mus
```

Results of pattern model for ratings

Call:

```
pattL.fit(obj = dat, nitems = 4, formel = ~age3, undec = TRUE)
```

```
Deviance: 622.3
log likelihood: -5509
eliminated term(s): ~age3
```

```
no of iterations: 36 (Code: 1 )
```

| | estimate | se | z | p-value |
|------------|----------|---------|---------|---------|
| clas | -0.31288 | 0.04074 | -7.680 | 0.0000 |
| coun | -0.22201 | 0.04012 | -5.534 | 0.0000 |
| rap | -0.50767 | 0.04310 | -11.779 | 0.0000 |
| clas:age32 | 0.14452 | 0.04834 | 2.990 | 0.0028 |
| coun:age32 | 0.17660 | 0.04799 | 3.680 | 0.0002 |
| rap:age32 | -0.22976 | 0.05407 | -4.249 | 0.0000 |
| clas:age33 | 0.66265 | 0.05159 | 12.845 | 0.0000 |
| coun:age33 | 0.75976 | 0.05317 | 14.289 | 0.0000 |
| rap:age33 | 0.04584 | 0.05679 | 0.807 | 0.4197 |
| U | 0.41961 | 0.01997 | 21.012 | 0.0000 |

7.5. Car configurator: Partial rankings, one subject covariate

Online configuration systems allow customers to actively participate in the creation of products and have become increasingly important in various market sectors. [Dabic and Hatzinger \(2009\)](#) report a study on car configurators that aimed at investigating the effects of certain person characteristics (such as gender) on the configuration process. Subjects were asked to configure a car according to their preferences. They could choose freely from several modules: such as exterior and interior design, technical equipment, brand, price, and producing country. The order of module choice was recorded as ranks. Since not all modules had to be chosen the response format was partial rankings. For illustrational purposes, we want to assess the effect of gender. First we fit the null model (without `sex`), but for later comparisons we have to set up the contingency table including `sex` through the `elim` argument (`formel = ~1` is default).

```
R> data("carconf")
R> mod.car.0 <- pattR.fit(carconf, 6, elim = ~ sex)
```

The model including `sex` is specified as

```
R> mod.car.sex <- pattR.fit(carconf, 6, formel = ~ sex, elim = ~ sex)
```

The deviance difference between the two models can be obtained using the log likelihoods of the models, i.e., $2 * (\text{mod.car.sex}\$ll - \text{mod.car.0}\$ll)$ and is 12.7 on $df = 5$. A worth plot (see [Figure 3](#)) is generated from

```
R> ws <- patt.worth(mod.car.sex)
R> colnames(ws) <- c("female", "male")
R> plot(ws, log = "y")
```

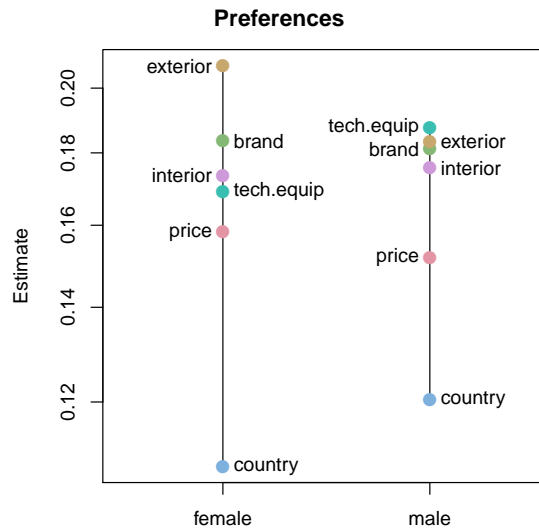


Figure 3: Worth plot for car features.

The worth parameters calculated from model `mod.car.sex` and displayed in Figure 3 show that males are most interested in technical equipment whereas females first look at exterior design. A thorough analysis is given in [Dabic and Hatzinger \(2009\)](#) where the other subject variables were also taken into account.

7.6. Attitudes towards foreigners: Paired comparisons, MNAR

Practical experience says that paired comparisons produce fewer missing observations than other response formats and thus are advantageous when investigating sensitive issues. [Weber and Hatzinger \(2011\)](#) conducted a survey, where 98 students compared four extreme statements about hypothetical consequences of migration through a paired comparison experiment (60 students completely refused to answer, the other 98 judged at least one paired comparison). Ties were allowed ('Don't know') and had been distinguished from missing responses ('Don't want to answer'). The variables and statements are

1. `crimRate`: Foreigners increase crime rates.
2. `position`: Foreigners take away training positions.
3. `socBurd`: Foreigners are a burden for the social welfare system.
4. `culture`: Foreigners threaten our culture.

We fit two models, one under MCAR assumption, the other specifying a nonignorable missing outcome model.

```
R> data("immig")
R> mod.MCAR <- pattPC.fit(immig, 4, undec = TRUE)
```

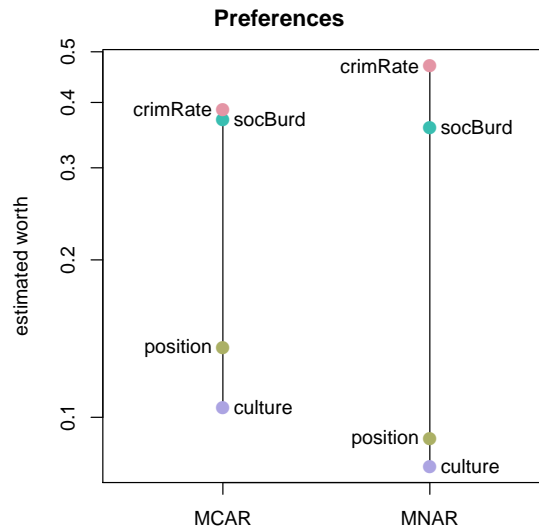


Figure 4: Worth plot for hypothetical migration threats under an MCAR and MNAR assumption.

```
R> mod.MNAR <- pattPC.fit(immig, 4, undec = TRUE, MISalpha = rep(TRUE, 4),
+   MISbeta = rep(TRUE, 4))
```

The different worth estimates under the two models can be seen from Figure 4.

7.7. Music style preference data: Ratings, latent classes

The function `pattnpml.fit` can be used to fit models for latent classes using a nonparametric random effects distribution. We apply the method to the previously used music style preference data. We first have to generate a design matrix. Missing values in the responses are not allowed, and we therefore carry out a complete case analysis.

```
R> dat <- na.omit(music[, c("clas", "coun", "rap", "conr", "age")])
R> dsg <- patt.design(dat, 4, resptype = "rating")
```

If we fit a model with $k = 1$ latent class, we have to specify the objects in the fixed effect formula term.

```
R> mod.lc <- pattnpml.fit(y ~ clas + coun + rap + conr, k = 1, design = dsg)
```

For $k > 1$ latent classes (and no fixed subject effects) the fixed part contains only the intercept whereas the objects are specified in the `random` argument. We fit models up to five latent classes.

```
R> mod.lc2 <- pattnpml.fit(y ~ 1, random = ~clas + coun + rap + conr,
+   k = 2, design = dsg, seed = 87654321)
R> mod.lc3 <- update(mod.lc2, k = 3)
R> mod.lc4 <- update(mod.lc2, k = 4)
R> mod.lc5 <- update(mod.lc2, k = 5)
```

To decide on the number of latent classes, we calculate the BIC for these models

```
R> BIC(mod.lc, mod.lc2, mod.lc3, mod.lc4, mod.lc5)
```

| | df | BIC |
|---------|----|----------|
| mod.lc | 4 | 915.2947 |
| mod.lc2 | 8 | 716.1136 |
| mod.lc3 | 12 | 655.9094 |
| mod.lc4 | 16 | 651.2091 |
| mod.lc5 | 20 | 659.4016 |

which indicates a 4-class solution (the minimal BIC).

Acknowledgments

Special thanks go to Brian Francis for detailed comments and discussions and also to Walter Katzenbeisser. Both are coauthors of several of our papers, who contributed to all the work that this article relies on. Particularly, we want to acknowledge Heather Turner and David Firth who dedicated a lot of work to the new `gnm` function to accelerate the fitting of highly complex PC models. We also want to thank two anonymous referees for their careful and detailed comments.

References

- Agresti A (2002). *Categorical Data Analysis*. John Wiley & Sons, New York.
- Aitkin MA (1996). “A General Maximum Likelihood Analysis of Overdispersion in Generalized Linear Models.” *Statistics and Computing*, **6**, 251–262.
- Aitkin MA, Francis B, Hinde J, Darnell R (2009). *Statistical Modelling in R*. Oxford University Press, Oxford.
- Critchlow DE, Fligner MA (1991). “Paired Comparison, Triple Comparison, and Ranking Experiments As Generalized Linear Models, and Their Implementation in GLIM.” *Psychometrika*, **56**, 517–533.
- Dabic M, Hatzinger R (2009). “Zielgruppenadäquate Abläufe in Konfigurationssystemen – Eine empirische Studie im Automobilmarkt – Partial Rankings.” In R Hatzinger, R Dittrich, T Salzberger (eds.), *Präferenzanalyse mit R: Anwendungen aus Marketing, Behavioural Finance und Human Resource Management*. Facultas, Wien.
- Davidson RR, Beaver RJ (1977). “On Extending the Bradley-Terry Model to Incorporate Within-Pair Order Effects.” *Biometrics*, **33**, 693–702.
- Dittrich R, Francis B, Hatzinger R, Katzenbeisser W (2006). “Modelling Dependency in Multivariate Paired Comparisons: A Log-Linear Approach.” *Mathematical Social Sciences*, **52**(2), 197–209.

- Dittrich R, Francis B, Hatzinger R, Katzenbeisser W (2007). “A Paired Comparison Approach for the Analysis of Sets of Likert Scale Responses.” *Statistical Modelling*, **7**(1), 3–28.
- Dittrich R, Francis B, Hatzinger R, Katzenbeisser W (2012). “Missing Observations in Paired Comparison Data.” *Statistical Modelling*, **12**(2), 117–143.
- Dittrich R, Hatzinger R, Katzenbeisser W (1998). “Modelling the Effect of Subject-Specific Covariates in Paired Comparison Studies with an Application to University Rankings.” *Applied Statistics*, **47**, 511–525.
- Dittrich R, Hatzinger R, Katzenbeisser W (2002). “Modelling Dependencies in Paired Comparison Experiments.” *Computational Statistics & Data Analysis*, **40**, 39–57.
- Einbeck J, Darnell R, Hinde J (2012). *npmlreg: Nonparametric Maximum Likelihood Estimation for Random Effect Models*. R package version 0.45-1, URL <http://CRAN.R-project.org/package=npmlreg>.
- Fienberg SE, Larntz K (1976). “Loglinear Representation for Paired and Multiple Comparison Models.” *Biometrika*, **63**, 245–254.
- Francis B, Dittrich R, Hatzinger R (2010). “Modeling Heterogeneity in Ranked Responses by Nonparametric Maximum Likelihood: How Do Europeans Get Their Scientific Knowledge?” *The Annals of Applied Statistics*, **4**(4), 2181–2202.
- Hatzinger R (2012). *prefmod: Utilities to Fit Paired Comparison Models for Preferences*. R package version 0.8-31, URL <http://CRAN.R-project.org/package=prefmod>.
- Hatzinger R, Francis BJ (2004). “Fitting Paired Comparison Models in R.” *Report 3*, Department of Statistics and Mathematics, Wirtschaftsuniversität Wien, Research Report Series. URL <http://epub.wu.ac.at/740/>.
- Kolb AY, Kolb DA (2005). “Learning Styles and Learning Spaces: Enhancing Experiential Learning in Higher Education.” *Academy of Management Learning & Education*, **4**(2), 193–212.
- Mallet A (1986). “A Maximum Likelihood Estimation Method for Random Coefficient Regression Models.” *Biometrika*, **73**, 654–656.
- R Development Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org/>.
- Rubin DB (1976). “Inference and Missing Data.” *Biometrika*, **63**, 581–592.
- Schöll B, Veith S (2011). *Learning Style Evaluation and Preferred Training Delivery Modes in Labour Market Training*. Master’s thesis, WU Wirtschaftsuniversität Wien. In German.
- Sinclair CD (1982). “GLIM for Preference.” In R Gilchrist (ed.), *Proceedings of the International Conference on Generalised Linear Models*, volume 14, pp. 164–178. Springer-Verlag.
- Strobl C, Wickelmaier F, Zeileis A (2011). “Accounting for Individual Differences in Bradley-Terry Models by Means of Recursive Partitioning.” *Journal of Educational and Behavioral Statistics*, **36**(2), 135–153.

- Thompson R, Baker RJ (1981). “Composite Link Functions in Generalized Linear Models.” *Applied Statistics*, **30**, 125–131.
- Turner H, Firth D (2011). *Generalized Nonlinear Models in R: An Overview of the **gnm** Package*. R package vignette, version 1.0-1, URL <http://CRAN.R-project.org/package=gnm>.
- Turner H, Firth D (2012). “Bradley-Terry Models in R: The **BradleyTerry2** Package.” *Journal of Statistical Software*, **48**(9). URL <http://www.jstatsoft.org/v48/i09/>.
- Weber D, Hatzinger R (2011). “A Novel Approach for Modelling Paired Comparisons Data with Non-Ignorable Missing Values on Students’ Attitudes Towards Foreigners.” *Data Analysis Bulletin*, **12**, 11–22.
- Wickelmaier F, Schmid C (2004). “A MATLAB Function to Estimate Choice Model Parameters from Paired-Comparison Data.” *Behavior Research Methods, Instruments, and Computers*, **36**(1), 29–40.

Affiliation:

Reinhold Hatzinger, Regina Dittrich
Institute for Statistics and Mathematics
WU Wirtschaftsuniversität Wien
Augasse 2–6
1090 Wien, Austria
E-mail: Reinhold.Hatzinger@wu.ac.at, Regina.Dittrich@wu.ac.at