The R package CUB: a class of mixture models for ordinal rating data

Maria Iannario University of Naples Federico II
Domenico Piccolo University of Naples Federico II
Rosaria Simone University of Naples Federico II

Abstract

The R package **CUB** has been developed for the analysis of ordinal rating data with a class of finite mixture distributions whose main feature is the probabilistic specification of the decision-making process as the combination of uncertainty and feeling components. Extensions include models for overdispersion, inflated categories and large heterogeneity occurrences. The parameter estimation procedure is based on maximum likelihood methods, with the implementation of the Expectation-Maximization (EM) algorithm. Several features of package **CUB**, including simulation routines, are presented on two data sets loaded within the package.

Keywords: Rating data, CUB models, CUBE models, *Shelter effect*, CUSH models, IHG models.

1. Introduction

Ordered categorical data are commonly used in scientific fields where respondents express a graduated evaluation on a specific item. Examples include ratings of preference in consumer studies, degree of pain measured on a visual analogue scale, numerical or verbal rating scale, sensory evaluation on food perception and appreciation, self evaluation of well-being, job satisfaction, economic perceived conditions and so on. Cumulative link models are a suitable class of models for such data since the ordered nature of observation is exploited and the flexible regression framework allows good fitting performances. These models with a logit link are widely known as the proportional odds model due to McCullagh (1980). The cumulative models are also known as ordinal regression models although the latter term is used to indicate other models for ordinal responses such as continuation ratio models and adjacent category models (see e.g. Agresti (2010); Tutz (2012)). Other alternatives with the indication of the logit and probit link functions are ordered logit models and ordered probit models (Greene and Hensher 2010).

Several R packages or functions for implementing these statistical models for ordered factor responses have been introduced, mainly addressing methods based on latent variables linked to observed categorical responses via threshold models. Among others, we quote **polr()** in **MASS** package (Venables and Ripley 2002), fitting logistic or probit regression models, for instance. In **rms** the function **lrm()** is available (Harrell 2009) for binary and proportional odds logistic regression; in **VGAM** the function **vgml()** (Yee 2010) to fit vector generalized linear models, whereas in **Zelig** it is possible to exploit **zelig()** to estimate several statistical models; in **nnet** the function **multinom()** allows to fit multinomial log-linear models (Ripley and Venables

2016) and in **Ordinal** the function **clm()** (Christensen 2015) fits general cumulative link models. For a Bayesian perspective, Markov Chain Monte Carlo for ordered probit regression are implemented in the package **MCMCpack** via calls to **MCMCoprobit()** (Martin *et al.* 2009).

An alternative approach is the use of models anchored to a structured discrete probability distribution (D'Elia and Piccolo 2005) with a close relationship to the data generating process. More specifically, a two-component mixture is designed to shape ordinal observations as a weighted combination of a preference component and an uncertainty component, each controlled by a given distribution.

The statistical appeal of these models relies on several features, among which: a sharp relationship with the data generating process producing ordinal data: parsimony in the parameter estimation; effective graphical devices useful for the interpretation, and direct inclusion of respondents' characteristics to specify how they relate with the two components.

This class of models stems from the baseline *C*ombination of a discrete *U*niform and a shifted *B*inomial random variable -CUB models- firstly introduced by Piccolo (2003) and D'Elia and Piccolo (2005), where two latent components, called *feeling* and *uncertainty*, are jointly considered to specify the mixture model. General references for the statistical background include Iannario (2012a,b, 2015); Iannario and Piccolo (2010a). A detailed list of alternative models is presented in Iannario and Piccolo (2016a); see also Tutz *et al.* (2016). Robustness issues for CUB models are discussed in Iannario *et al.* (2016b).

Further methodological studies have been published about CUB models, among others Corduas *et al.* (2009); Gambacorta *et al.* (2014); Grilli *et al.* (2014) for a latent class version; Iannario (2012c,d, 2014); Manisera and Zuccolotto (2014a); Iannario *et al.* (2016a) for including *don't know* options also with covariates or for assuming non-linear spacing between adjacent categories: Manisera and Zuccolotto (2014b). In addition, applications have been performed in many disciplines, ranging from labour economics (Gambacorta and Iannario 2013) to demography (Iannario and Piccolo 2010b), happiness economics (Capecchi and Piccolo 2014), linguistics (Balirano and Corduas 2008), marketing (Iannario *et al.* 2012), medicine (D'Elia 2008), sensory science (Piccolo and D'Elia 2008; Capecchi *et al.* 2015). Proposals for the generalization of the CUB model in a bivariate perspective are discussed by Corduas (2015). Moreover, Andreis and Ferrari (2013), Colombi and Giordano (2016) introduced a general multivariate setting.

This article deals with the package **CUB** written in the R environment and available from CRAN at `http://CRAN.R-project.org`. It is presented in its first consistent update, but the implemented program has been long experienced among scholars and researchers working with CUB models and their extensions.

The paper is organized as follows. Section 2 briefly reviews *GE*neralized *M*ixture Models with uncertainty (GEM) for ordinal data stemming from CUB models. In Section 3 the main functions and features of package **CUB** are illustrated by means of case studies based on two data-sets loaded within the package. Finally, extra features of GEM models and future extensions are briefly discussed in Section 4.

Ntice that package **CUB**, although performs statistical analysis for the whole class of GEM models, is named after the original proposal.

2. GEM models specification

The basic idea of GEM framework is to model the ordinal responses $(R_1, \ldots, R_n)$, where R_i is given by the i-th subject, $i = 1, \ldots, n$, with respect to a given item I, on the basis of two main components related to the decisional process. The first component (C_1) related to the feeling (attraction, satisfaction, awareness towards the item): in the baseline CUB model, it is modelled by means of the shifted Binomial random variable. The second one (C_2) concerns the uncertainty (indecision, blurriness, fuzziness) and it is modelled by means of a discrete Uniform distribution over the support $\{1, \ldots, m\}$. Here and in the sequel, m will denote the number of ordinal categories, and it is assumed that $m > 3$ for identifiability purposes (Iannario 2010). All available information on subjects' characteristics (covariates) are collected in a matrix $\boldsymbol{T}$, and $\boldsymbol{Y}$ and $\boldsymbol{W}$ are submatrices of $\boldsymbol{T}$ reporting values of respondents' covariates useful to explain uncertainty and feeling components, respectively.

Let $\boldsymbol{\theta} = (\boldsymbol{\beta}', \boldsymbol{\gamma}')'$ be the parameter vector characterizing the distribution of $(R_1, \ldots, R_n)$, with $\boldsymbol{\beta}', \boldsymbol{\gamma}'$ denoting the parameter vector for the uncertainty and feeling components, respectively. Then, the mixture regression model has the following form:

$$Pr\left(R_i = r \mid \boldsymbol{y}_i, \boldsymbol{w}_i, \boldsymbol{\theta}\right) = \pi_i \, Pr\left(C_{1i} = r \mid \boldsymbol{w}_i\right) + \left(1 - \pi_i\right) Pr\left(C_{2i} = r\right), \tag{1}$$

for $i = 1, \ldots, n$ and $j = 1, \ldots, m$, where $\pi_i = \pi(\boldsymbol{y}_i, \boldsymbol{\beta}) \in (0, 1]$ are introduced to weight the two components and $\boldsymbol{w}_i \in \boldsymbol{W}$, $\boldsymbol{y}_i \in \boldsymbol{Y}$ include the selected covariates for the i-th subject. They are related to the parameters by means of a logit link, a not compulsory but preferred choice for easiness of interpretation and robustness properties (Iannario *et al.* 2017):

$$logit\left(1 - \pi_i\right) = -\boldsymbol{\beta}' \, \boldsymbol{y}_i; \qquad logit\left(1 - \xi_i\right) = -\boldsymbol{\gamma}' \, \boldsymbol{w}_i. \tag{2}$$

The implementation of GEM models relies on the **Formula** interface (Zeileis and Croissant 2010), allowing to specify different regressor matrices for the different parameters. The following call will estimate a GEM model for given `ordinal` observations:

```
GEM(Formula, family, ...)
```

where `Formula` is an object of class **Formula**, and `family` indicates which sub-class of models has to be fitted. Optional inputs for the specification of the models can be passed via the `...` argument, which serves also to modify some technical parameters needed for the estimation procedure. For instance, it is possible to change the maximum number of iterations allowed for running the optimization algorithm (`maxiter=500` by default) and the error tolerance for final estimates (`toler=1e-6` by default). The argument `data` is optional. The number of categories `m` is, in general, internally retrieved but it is advisable to pass it as an argument to the call if some category has zero frequency.

As mentioned, the benchmark model is the CUB mixture model, where the preference part is modelled according to a shifted Binomial random variable:

$$b_r\left(\xi_i\right) = \binom{m-1}{r-1} \xi_i^{m-r} \left(1 - \xi_i\right)^{r-1}, \qquad r = 1, 2, \ldots, m$$

with feeling parameters given by ξ_i, $i = 1, \ldots, n$, as in (2).

The probability distribution chosen to model uncertainty is the discrete Uniform, thus it does not imply estimable parameters ensuring model parsimony. An increase of π_i implies a reduced

impact of the uncertainty component; thus, the quantity $(1 - \pi_i)$ is a (normalized) measure of the uncertainty implied by the model. Alternative subjective choices of distributions for this latent component are proposed and motivated in Gottard *et al.* (2016).

In the simplest version, the GEM model works on the ratings given to a single item and provides a global measure of feeling and uncertainty for the whole population under study by letting $\pi_i = \pi$ and $\xi_i = \xi$, that is, by considering no knowledge about subjects' characteristics. The corresponding probability distribution can be computed via `probcub00(m, pai, csi)`.

Other extensions which are implemented within package **CUB** concern the ability to take into account the presence of inflated categories and the overdispersion effect.

2.1. Inflated CUB models

Inflated CUB models imply the presence of a *shelter* effect (Iannario 2012a) located at category $s \in \{1, \ldots, m\}$.

CUB models are thus extended with the introduction of a degenerate random variable $D_r^{(s)} = I(R = s)$, whose probability mass is concentrated at $r = s$, as a third component in (1). The function returns the estimation of parameters related both to the specification:

$$Pr\,(R = r \mid \boldsymbol{\theta}) = \pi_1\, b_r(\xi) + \pi_2\, \frac{1}{m} + (1 - \pi_1 - \pi_2)\, D_r^{(s)}\,, \tag{3}$$

and to the alternative formulation:

$$Pr\,(R = r \mid \boldsymbol{\theta}) = \delta \left[D_r^{(s)} \right] + (1 - \delta) \left[\pi^*\, b_r(\xi) + (1 - \pi^*)\, \frac{1}{m} \right], \quad r = 1, 2, \ldots, m. \tag{4}$$

They are equivalent specifications since:

$$\begin{cases} \pi_1 = \pi^*(1 - \delta) \\ \pi_2 = (1 - \pi^*)(1 - \delta) \end{cases} \iff \begin{cases} \pi^\star = \dfrac{\pi_1}{\pi_1 + \pi_2} \\ \delta = 1 - \pi_1 - \pi_2. \end{cases}$$

The corresponding probability distributions can be computed via `probcubshe1(m, pai1, pai2, csi, shelter)` for (3) and `probcubshe2(m, pai, csi, delta, shelter)` for (4), respectively.

For the parameterization (4), the estimable parameter vector is $\boldsymbol{\theta} = (\pi^*, \xi, \delta)'$; thus it is immediate to quantify the *shelter effect* by means of the parameter δ. Moreover, the modification of the uncertainty component so induced is evaluated by comparing the π parameter in model (1) with π_1 in model (3). A further specification of the *shelter effect* adheres to the so-called *satisficing behaviour*, impletemented via `probcubshe3(m, lambda, eta, csi, shelter)`. Covariates in the model can be specified for all parameters by considering generalized CUB models (G*e*CUB) (Iannario and Piccolo 2016b).

The examination of real phenomena where responses are affected by an uncertainty level close to the maximum and with inflation in just one category suggested the introduction of CUSH models (*C*ombination of a *U*niform distribution and a *SH*elter component) (Capecchi *et al.* 2015; Capecchi and Piccolo 2016; Capecchi and Iannario 2016), whose distribution is especially useful in dealing with sample surveys affected by large heterogeneity (several examples are observed in the analysis of activities for leisure times, see the discussion below). In other

words, CUSH models are nested into GeCUB ones when the shifted Binomial component tends to have zero weight (Capecchi and Iannario 2016): indeed, the distribution without covariates may be obtained by (4) when $\pi^\star = 0$. The routine `probcush(m, delta, shelter)` computes the corresponding probability distribution:

$$Pr\left(R = r \mid \boldsymbol{\theta}\right) = \delta\ D_r^{(s)} + (1-\delta)\,\frac{1}{m}, \qquad r = 1, 2, \ldots, m, \tag{5}$$

where $\boldsymbol{\theta} = \delta$, the weight of the *shelter effect*. If $\boldsymbol{X}$ is a submatrix of $\boldsymbol{T}$ specifying the observations on selected covariates for explaining the *shelter effect*, then the link: $logit(\delta_i) = \boldsymbol{\omega}^{'}\boldsymbol{x}_i, i = 1, \ldots, n$, will specify parameters for each respondent profile (here, $\boldsymbol{\omega}^{'}$ includes an intercept term).

2.2. Overdispersion

CUB models are not suitable to account for a possible extra-variability among responses, since location and variability of a Binomial random variable are jointly and severely constrained. Thus, a parsimonious extension has been proposed by introducing CUBE models (Iannario 2014, 2015). Here, the shifted Binomial distribution for $Pr\left(C_{1i} = r\right)$ is replaced by a shifted Beta-Binomial distribution $g_r(\xi, \phi)$ over the support $\{1, \ldots, m\}$:

$$g_r(\xi,\phi) = \binom{m-1}{r-1} \frac{\prod_{k=1}^{r}\left(1-\xi+\phi\,(k-1)\right)\prod_{k=1}^{m-r+1}\left(\xi+\phi(r-1)\right)}{(1-\xi+\phi\,(r-1))(\xi+\phi\,(m-r))\prod_{k=1}^{m-1}(1+\phi\,(k-1))}, \quad r = 1, 2, \ldots, m, \tag{6}$$

(see `betar()`). Thus, the probability distribution of a CUBE model -computed by `probcube(m, pai, csi, phi)`- is given by:

$$Pr\left(R = r \mid \boldsymbol{\theta}\right) = \pi\, g_r(\xi,\phi) + (1-\pi)\,\frac{1}{m} \qquad r = 1, 2, \ldots, m. \tag{7}$$

The additional parameter ϕ which characterizes the distribution is a direct measure of overdispersion (Iannario 2016). When $\phi = 0$, a CUBE model reduces to a CUB one; thus, the latter is nested into the first one: this implies that the improvement of the fit of CUBE over CUB models for given data can be easily established by likelihood ratio tests.

Covariates may be introduced for explaining all the three parameters or only the feeling component of the CUBE mixture (Piccolo 2015). The logarithmic link is customarily used for the overdispersion parameter: $\log\left(\phi_i\right) = \boldsymbol{\alpha}^{'}\boldsymbol{z}_i$ where $\boldsymbol{z}_i \in \boldsymbol{Z}$, and $\boldsymbol{Z}$ is a submatrix of $\boldsymbol{T}$ ($\boldsymbol{\alpha}^{'}$ includes an intercept term). Feeling and uncertainty components are related to subjects' characteristics as in (2).

It has been remarked (Iannario 2012b) that CUBE models with:

$$\pi = 1; \qquad \xi = \frac{(m-1)\theta}{1+m(\theta-2)}; \qquad \phi = \frac{1-\theta}{1+\theta(m-2)}, \tag{8}$$

correspond to IHG models (D'Elia 2003) with $\theta \in [0, 1]$, where IHG stands for Inverse Hypergeometric distribution. These models arise from unimodal extreme-feeling distributions with no uncertainty. In this case, for the i-th respondent, a parameter θ_i characterizes the

model and gives the probability of observing a rating corresponding to the first/last category: therefore, θ_i is a direct measure of preference, attraction, pleasantness towards the investigated item. The routine `probihg(m, theta)` computes the IHG probability distribution over m categories:

$$Pr\left(R = r \mid \boldsymbol{\gamma}\right) = \theta(1-\theta)^{r-1}\,\frac{m-1}{m}\prod_{l=1}^{r}\frac{m-l+1}{m-l+\theta(l-1)} \qquad r = 1,\ldots,m.$$

Also for this sub-class of models, it is possible to study response patterns among respondents by including subjects' covariates to explain the preference parameter θ_i, $i = 1,\ldots,n$, via a logit link: $logit(\theta_i) = \boldsymbol{\nu}'\boldsymbol{u}_i$, where $\boldsymbol{u}_i \in \boldsymbol{U}$ and $\boldsymbol{U}$ is a submatrix of $\boldsymbol{T}$ ($\boldsymbol{\nu}'$ includes an intercept term).

After this self-contained overview of the statistical methodology grounded on GEM models, next section provides guidelines for the main calls to fit GEM models on given data when using package **CUB**.

2.3. Main calls

As already specified, the baseline GEM model is the CUB distribution. To fit a CUB model to given ordinal data arranged in a vector `ordinal`, the most general call to run is:

```
GEM(Formula(ordinal~Y|W|X), family="cub", ...)
```

where `Y, W, X` indicate the model matrices of explanatory variables for uncertainty, feeling and possible *shelter* components, respectively. If, for all components, no covariate is included in the model, then `Y=0`, `W=0`, `X=0` need to be specified. For instance

```
GEM(Formula(ordinal~0|0|0), family="cub")
```

fits a CUB model with no covariates for given `ordinal` data. Then,

```
GEM(Formula(ordinal~Y|0|0), family="cub")
```

considers only the covariate matrix `Y` to explain the uncertainty component. Similarly,

```
GEM(Formula(ordinal~0|W|0), family="cub")
```

includes the covariate matrix `W` to explain only the feeling component, whereas:

```
GEM(Formula(ordinal~Y|W|0), family="cub")
```

specifies a CUB model with covariates for both components.

To test for a *shelter effect*, it is possible to implement either a CUB model without covariates by specifying the value of the *shelter* s in the main call to `GEM()`:

```
GEM(Formula(ordinal~0|0|0), family="cub", shelter=s)
```

or to implement the more comprehensive specification concerning GeCUB models:

```
GEM(Formula(ordinal~Y|W|X), family="cub", shelter=s)
```

CUSH models can be fitted by the main function `GEM()`, optionally with the specification of covariates, by specifying `family` argument to `"cush"`:

```
GEM(Formula(ordinal~0), family="cush", shelter = s)      # without covariates
GEM(Formula(ordinal~X), family="cush", shelter = s)      # with covariates
```

If `Y`, `W` and `Z` are matrices of values of selected covariates for explaining uncertainty, feeling and overdispersion, respectively, CUBE models can be estimated via:

```
GEM(Formula(ordinal~0|0|0), family="cube")     # without covariates
GEM(Formula(ordinal~0|W|0), family="cube")     # with covariates for feeling
GEM(Formula(ordinal~Y|W|Z), family="cube")     # with covariates for all param-
eters
```

For CUBE model with no covariates, the default option `expinform = FALSE` implies that the variance-covariance matrix of estimates is based on the observed information matrix. If `expinform = TRUE` is added as an extra argument, the procedure considers the expected information matrix instead. Note that the fitting procedure for CUBE models with covariates for all components might be particularly slow: hence, in the first runs, we tentatively suggest to lower the tolerance and the maximum number of iterations allowed for the optimization algorithm by specifying `maxiter` and `toler` arguments in the function call (for instance, `maxiter=50`, `toler=1e-3`). Alternatively, one might estimate a random sub-sample of data and adopt the resulting estimates as preliminary values for a GEM fit to the whole data.

Finally, estimation of IHG models is implemented via:

```
GEM(Formula(ordinal~0),family="ihg")          # without covariates
GEM(Formula(ordinal~U),family="ihg")          # with covariates
```

Every call to the main `GEM` creates an object of the class `"GEM"` -to which the subclass specified by the `family` argument is appended. Next sections provide details on inferential issues for GEM models, and the corresponding implemented `S3` methods.

3. Implementation and inference in CUB

Package **CUB** fits the models presented in Section 2 using Maximum Likelihood (ML) estimation as performed by classical EM procedures (McLachlan and Krishnan 1997). For every unspecified detail about GEM models, see Iannario and Piccolo (2016a). Further, details about the EM steps for CUB models can be found in Piccolo (2006), for CUBE models in Iannario (2014); Piccolo (2015), and for GeCUB models in Iannario and Piccolo (2016b). For CUSH

models, the reader is referred to Capecchi and Piccolo (2016) for a full discussion, whereas for IHG see D'Elia (2003).

3.1. Method implementation

For an object of class `"GEM"`, storing the information of the fitting procedure of a GEM model to given data, the standard `S3` method `logLik()` applies to return the maximized log-likelihood at the final ML estimates. Thus, in the baseline version of a GEM model, parameters are estimated by maximizing the log-likelihood:

$$\ell(\boldsymbol{\theta}) = \sum_{i}^{n} \log \left[Pr\left(R_i = r \mid \boldsymbol{y}_i, \boldsymbol{w}_i, \boldsymbol{\theta}\right) \right].$$

The optimization procedure is run via the function `optim()`, except for CUB models without covariates (for which the EM steps are directly implemented: D'Elia and Piccolo (2005)) and for CUSH models without covariates (for which the ML estimate of the *shelter* parameter can be straightforwardly computed, see Capecchi and Piccolo (2016)). Whenever bounds on parameter estimates have to be required, the box-constrained option for `optim()` is selected, and it considers a restriction of the admissible range of the parameter space: in this case, `method = "L-BFGS-B"` is selected for multidimensional optimization problems (with finite difference approximation selected with the option `gr = NULL`), while `method = "Brent"` is selected for the unidimensional optimization problem when IHG models with no covariate are fitted. In all other cases, the default Nelder-Mead method is selected.

In addition, for given ordinal data and model-specific parameters, log-likelihood values can be computed via the following main calls (the default option is that neither covariate nor *shelter effect* are included):

```
llCUB  <- loglikCUB(ordinal,m,param,Y=Y,W=W,X=X,shelter=s)
llCUBE <- loglikCUBE(ordinal,m,param,Y=Y,W=W,Z=Z)
llIHG  <- loglikIHG(ordinal,m,param,U=U)
llCUSH <- loglikCUSH(ordinal,m,param,X=X,shelter=s)
```

The vector `param` lists the parameters for the specified model, with the uncertainty parameters always specified first, followed by the feeling ones and, last, by the overdispersion or *shelter* parameters when considering CUBE or CUB models with *shelter effect*, respectively. For IHG and CUSH models, `param` is a one-dimensional vector when no covariates are considered, otherwise it lists all the regression coefficients for the preference and the *shelter* parameter, respectively. When covariates are included in the model, the first component of the corresponding (sub)vector is the intercept term (not to be included in the matrix), followed by as many parameters as the number of covariates used to explain the given component.

The variance-covariance matrix of final ML estimates for a fitted model can be retrived by applying the standard `S3` method `vcov()` to an object of class `GEM`. For testing IHG models with or without covariates, for CUSH models with covariates as well as for CUBE models with covariates for all the three parameters, the option `hessian = TRUE` is selected when calling `optim()` to return a numerically differentiated Hessian matrix at the final estimates. For the other models, the code implements specific inferential results, as the explicit formulae of

the observed and expected variance-covariance matrices (Piccolo 2006, 2015; Iannario 2012a, 2014).

To compute the (observed) variance-covariance matrix corresponding to given ordinal data and model-specific parameters (for CUB and CUBE models), the following functions are available:

```
varCUB <- varmatCUB(ordinal, m, param, Y = Ycovar, W = Wcovar, X = Xcovar,
                    shelter = shelter)
```

```
varCUBE <- varmatCUBE(ordinal, m, param, Y = Ycovar, W = Wcovar, Z = Zcovar,
                      expinform = FALSE)
```

No covariate is considered neither for `varmatCUB()` nor for `varmatCUBE()` by default. In addition, default options for `varmatCUB` do not consider *shelter effect*. For `varmatCUBE()`, the default option `expinform = FALSE` provides the variance-covariance matrix based on the observed information matrix. Setting `expinform = TRUE`, the variance-covariance matrix will be based on the expected information matrix instead (as advisable when running simulation studies).

Other standard S3 methods implemented for `GEM` objects include:

- `coef()` to retrieve final parameter estimates;
- `logLik()` to obtain the maximized log-likelihood at the final ML estimates;
- `BIC()` to compute the BIC index of the fitted model (in order to compare non-nested models (Schwarz 1978));
- `fitted()` to display fitted probability distibution.
- `summary()` to display summary results of the fitting procedures:
    1. ML parameter estimates of the fitted model, asymptotic standard errors and Wald-tests (based on the asymptotic variance-covariance matrix as returned by `vcov()`);
    2. For testing purposes, a list of likelihood-based measures (Log-likelihood, Mean Log-likelihood, and for models without covariates also Log-likelihood of the saturated model, Deviance, Log-likelihood of the Uniform and shifted Binomial).
    3. BIC, AIC (Akaike 1974) and ICOMP (Bodzogan 1990) measures for the fitted model, in order to compare non-nested models.

Every call to the main function `GEM()` stores the following slots to record some further fitting information:

- `$time`: the time employed for the estimation procedure;
- `$niter`: the number of iterations required to the EM algorithm for convergence within the fixed tolerance.
- `$formula`: to retrieve the Formula object given as input.

Parameter correlation matrix corresponding to an object of class `GEM` can be computed via `cormat()`, or printed out by specifying `correlation=TRUE` when calling `summary()`.

Table 1 summarizes the main calls to function `GEM` to fit the different models. Possible extra arguments to include in the call will be further illustrated and discussed along with the examples.

3.2. Plot facilities

One of the advantageous features of GEM models is that they allow for easy interpretation of parameters. This is especially convincing for models with no covariates or for (at most) one dichotomous variable, a case for effective graphical devices.

For an object of class `GEM`, the generic function `makeplot(object)` will return a plot which compares fitted probabilities and observed relative frequencies. Moreover, for CUB models, given the one-to-one correspondence between the probability distribution of R and the parameter vector $\boldsymbol{\theta} = (\pi, \xi)'$, a relevant feature of the approach is the possibility to visualize a CUB model as a point in the parameter space, that is the unit square. To this scope, the user is referred to functions `cubvisual`(), `multicub`() and `cubshevisual`(). Similar features are performed by `cubevisual`() and `multicube`() for CUBE models.

Other instances of graphical analysis will be shown along with discussion of empirical evidence in Section 4.

4. Package CUB in use

Some basics routines that can help in the comprehension of CUB models and their extensions are presented. The following code produces Figure 4.1, in which the plots of CUBE probability mass functions for varying parameters are shown, as in Iannario (2015). Figure 4.1 displays also the expectation and the variance of the given CUBE distributions, computed via `expcube()` and `varcube()`, respectively. Similar functions for CUB models are `expcub00()` and `varcub00()`.

```
#############################################################################
### Selection of 9 CUBE models with csi = 0.3 over 9 ordinal categories  ***
#############################################################################
m<-9; csi<-0.3
########### varying pai and phi parameters
paival<-seq(0.9,0.1,by=-0.1)
phival<-rep(c(0.05,0.1,0.3),times=3)
model<-cbind(paival,phival); nmodels<-nrow(model)
#########################################
par(mfrow = c(3,3))
par(mar = c(2,4,3,1)+0.1)
### Probability distribution plots for jmod=1,2,...,9
for (jmod in 1:nmodels){
  paij<-model[jmod, 1]
        phij<-model[jmod, 2]
```

Model	Call
CUB models	
with no covariates	`GEM(Formula(ordinal~0|0|0),family="cub")`
with no covariates and shelter effect	`GEM(Formula(ordinal~0|0|0),family="cub",shelter=shelter)`
with covariate matrix `Y` for uncertainty	`GEM(Formula(ordinal~Y|0|0),family="cub")`
with covariate matrix `W` for feeling	`GEM(Formula(ordinal~0|W|0),family="cub")`
with covariate matrices `Y` and `W` for uncertainty and feeling	`GEM(Formula(ordinal~Y|W|0),family="cub")`
with covariate matrices `Y`, `W`, `X` for uncertainty, feeling and shelter effect	`GEM(Formula(ordinal~Y|W|X),family="cub")`
CUBE models	
with no covariates	`GEM(Formula(ordinal~0|0|0),family="cube")`
with covariate matrix `W` for feeling	`GEM(Formula(ordinal~0|W|0),family="cube")`
with covariate matrices `Y`,`W` and `Z` for uncertainty, feeling and overdispersion	`GEM(Formula(ordinal~Y|W|Z),family="cube")`
CUSH models	
with no covariates	`GEM(Formula(ordinal~0),family="cush",shelter=shelter)`
with covariate matrix `X` for shelter effect	`GEM(Formula(ordinal~X),family="cush",shelter=shelter)`
IHG models	
with no covariates	`GEM(Formula(ordinal~0),family="ihg")`
with covariate matrix `U` for the preference parameter	`GEM(Formula(ordinal~U),family="ihg")`

Table 1: Summary of possible calls to `GEM()`

```
        prob<-probcube(m, paij, csi, phij)
        exp<-expcube(m, paij, csi, phij)
        var<-round(varcube(m, paij, csi, phij), digits = 2)
        plot(1:m, prob, type = "h", lwd = 3, ylim = c(0,0.4), xlab = "",
       ylab = "Pr(R=r)", las = 1)
        text(2.3, 0.38, bquote(pi == .(paij)), cex = 0.7)
  text(5, 0.38, bquote(xi == .(csi)), cex = 0.7)
  text(7.8, 0.38, bquote(phi == .(phij)), cex = 0.7)
  text(7, 0.285, bquote(sigma^2 == .(var)), cex = 0.7)
  text(3, 0.28, bquote(mu == .(exp)), cex = 0.7)
}
par(mar = c(5,4,4,2)+0.1)              ### reset standard margins
par(mfrow = c(1,1))                    ### reset plot screen
```

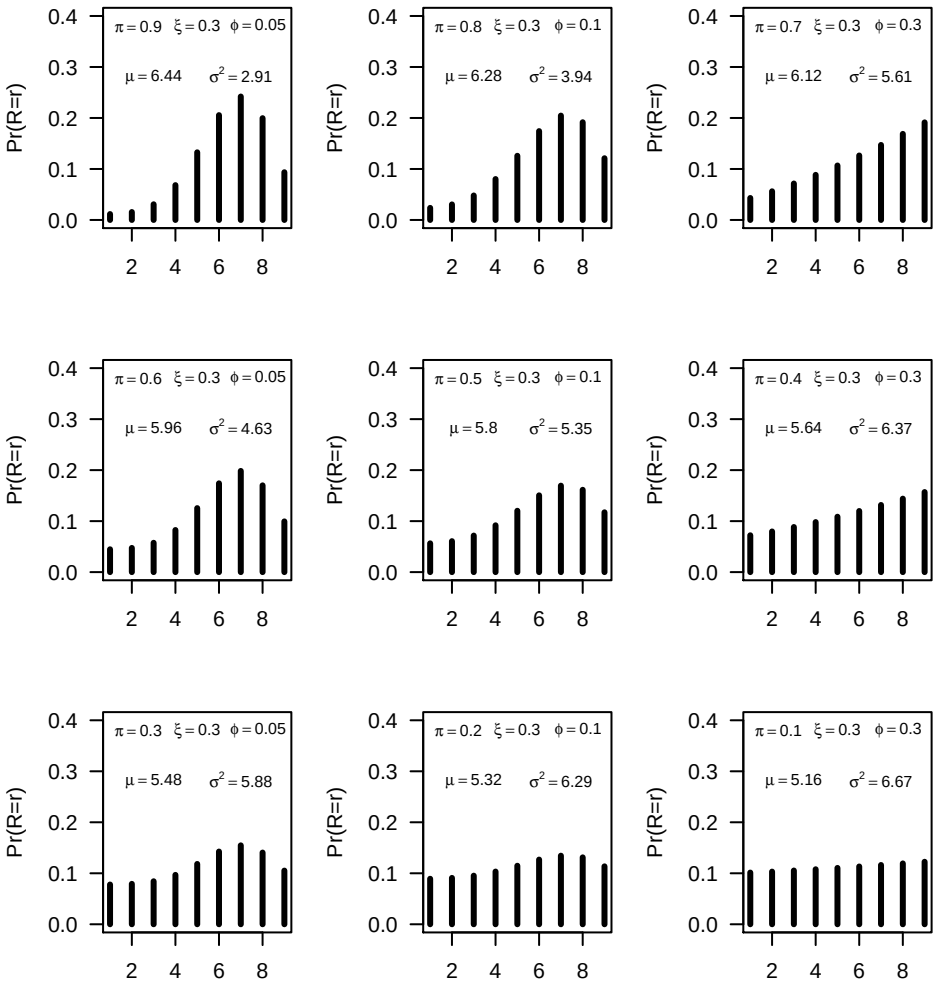


Figure 4.1: Probability distributions of CUBE models.

In the following subsections we discuss the usage of package **CUB** by means of two real data sets, `univer` and `relgoods`, bundled within the package. Detailed descriptions, related sample experiment and further references can be found at http://www.labstat.it/home/research/resources/cub-data-sets-2/.

4.1. CUB models

Data set `univer` arises from a sample survey on students' evaluation of the Orientation services that has been administed across all the Faculties of University of Naples Federico II in five waves. Participants were asked to express their ratings on a 7 point Kikert-type scale (1 = "very unsatisfied", 7 = "extremely satisfied") on the following items:

- `informat`: Level of satisfaction about the acquired information
- `willingn`: Level of satisfaction about the willingness of the staff
- `officeho`: Level of satisfaction about the opening hours
- `competen`: Level of satisfaction about the competence of the staff
- `global`: Level of global satisfaction

Data set collected in 2002 consists of 2179 observations: the first 7 columns correspond to subjects covariates (for instance, the dichotomous variable `gender`, equal to 0 for men and to 1 for women, and the continuous measurement `age`).

As a first step, we show how to visualize simultaneously the ordinal variables included in the data set `univer` by means of the function `multicub()`: it fits a CUB model to every ordinal variable. Then each model is represented as a point in the parameter space corresponding to the obtained ML uncertainty $1 - \hat{\pi}$ and feeling $1 - \hat{\xi}$ estimates: indeed, estimable parameters are π and ξ, but interpretation depends on the orientation of the measurement scale. If the evaluation is positive in the direction of the scale, then the actual measure of feeling is $1 - \xi$; flag arguments `csiplot` and `paiplot` may be switched to `TRUE` if direct parameter visulization is preferred. Notice that `multicub()` allows for a comparative visual analysis of vectors of rating data, possibly with different lengths and also over different numbers of categories: for this reason, they are required to be stored in a list when calling `multicub()` and optionally a vector `mvett` can be set as input argument to specify the corresponding scale lengths. If only one ordinal variable is considered, then the same visual analysis is implemented via `cubvisual()`.

```
data(univer)
listord<-univer[,8:12] # only ratings, excluding covariates
labels<-names(univer)[8:12]
multicub(listord, labels = labels,
        caption ="CUB models on Univer data set", pch = 19,
        pos = c(1,rep(3, ncol(listord)-1)),ylim=c(0.75,1),xlim=c(0,0.4))
```

From Figure 4.2, we may conclude that the highest feeling has been expressed for the willingness of the staff, the lowest for the scheduled office hours. However, since the latter item is

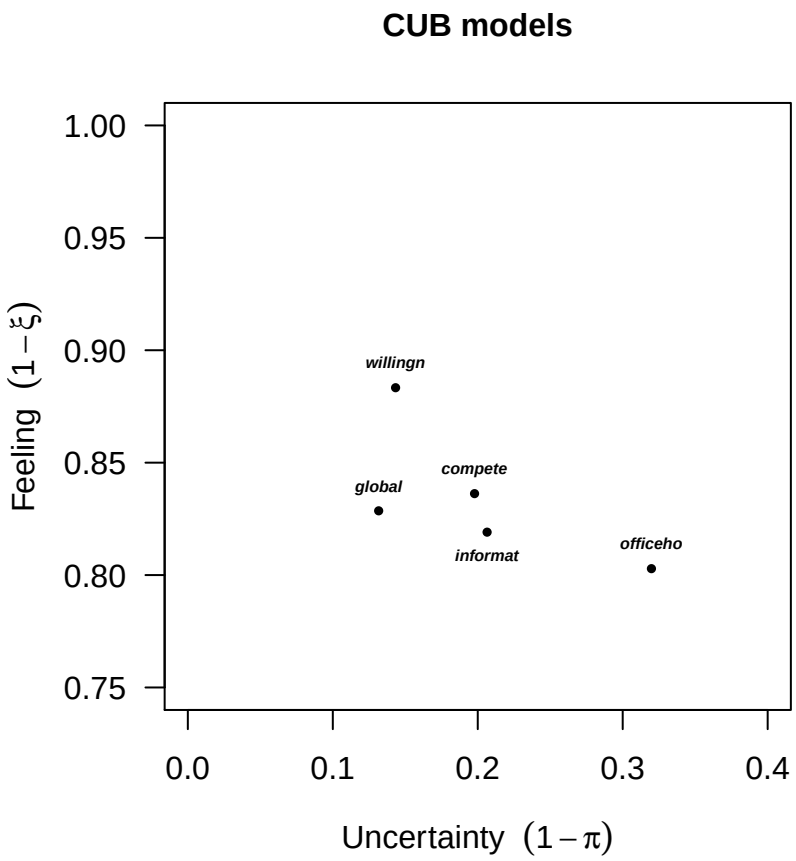


Figure 4.2: CUB models without covariates for the items of `univer` ($m = 7$).

affected by the highest uncertainty, it deserves further specification; thus, hereafter we focus on the item `officeho`.

```
## CUB model without covariates for "officeho"
cub_00<-GEM(Formula(officeho~0|0|0), family="cub",data=univer)
summary(cub_00,digits=5)

=============================================================================
=====>>> CUB  model    <<<=====   ML-estimates via E-M algorithm
=============================================================================
 m= 7  Sample size: n= 2179  Iterations= 28  Maxiter= 500
=============================================================================
Uncertainty
    Estimates   StdErr   Wald
pai   0.68019 0.019349 35.153
=============================================================================
Feeling
```

```
    Estimates    StdErr   Wald
csi   0.19714 0.0058822 33.515
======================================================================
Log-lik            = -3759.917
Mean Log-likelihood= -1.72552
Log-lik(UNIFORM)   = -4240.138
Log-lik(saturated) = -3706.922
Deviance           = 105.9914
----------------------------------------------------------------------
AIC       = 7523.834
BIC       = 7535.208
ICOMP     = 7521.135
======================================================================
Elapsed time= 0.02 seconds =====>>> Thu Feb 08 19:13:18 2018
======================================================================
```

The estimated CUB model is identified with uncertainty and feeling given by:

```
param<-coef(cub_00,digits=3)
param


pai 0.680
csi 0.197

uncertainty<-1-param[1]
uncertainty

[1] 0.32

feeling<-1-param[2]
feeling

[1] 0.803
```

The following call:

```
## CUB model without covariates
makeplot(cub_00)
```

provides a plot showing both the observed relative frequencies and the fitted probabilities, as in Figure 4.3 (left panel).

Accurate initial values for parameters allow to reach convergence to the ML solutions in acceptable time (Iannario 2012c): in this regard, the EM algorithm for CUB models without covariates is initialized within `GEM(Formula,family="cub")` by means of `inibest(m,freq)`,

a routine that returns preliminary parameter estimates of a CUB model for a given frequency distribution:

```
data(univer)
freq<-tabulate(univer$officeho,nbins = 7)
ini<-inibest(m,freq) # preliminary estimates for c(pai,csi)
ini



pai 0.9415651
csi 0.2777778
```

Features of `inibest(m,freq)` can be exploited to obtain preliminary parameter estimates for any purpose. Similarly, `inigrid()` runs a grid search across the parameter space and it returns the parameter vectors corresponding to the maximum of log-likelihood on the given grid.

In order to improve the fit and interpretation of data, we introduce covariates in the model. Specifically, to explain the feeling component we consider the dichotomous covariate `freqserv`, indicating the usage frequency of the service with levels 0 and 1 for non-regular and regular users.

```
cub_csi<-GEM(Formula(officeho~0|freqserv|0), family="cub",data=univer)
summary(cub_csi,digits=3)

==========================================================================
=====>>> CUB  model    <<<=====   ML-estimates via E-M algorithm
==========================================================================
 m= 7  Sample size: n= 2179  Iterations= 24  Maxiter= 500
==========================================================================
Uncertainty
    Estimates StdErr Wald
pai     0.687 0.0192 35.9
==========================================================================
Feeling
         Estimates StdErr   Wald
constant    -1.152 0.0403 -28.58
freqserv    -0.811 0.0850  -9.55
==========================================================================
Log-lik            = -3704.357
Mean Log-likelihood= -1.7
--------------------------------------------------------------------------
AIC       = 7414.714
BIC       = 7431.773
ICOMP     = 7410.96
==========================================================================
Elapsed time= 0.5 seconds =====>>> Thu Feb 08 19:13:19 2018
==========================================================================
```

For the previous example, the regression coefficients for the logistic link (all of which are significant) can be retrieved by:

```
gama<-coef(cub_csi)[2:3]
gama

[1] -1.151942 -0.811228

gama0<-gama[1]  ## intercept term
gama1<-gama[2]
```

Then, to compute the feeling parameter corresponding to non-regular and regular users according to (2), respectively, it is possible to run:

```
csi_nru <- logis(0, gama)   ## csi parameter for non regular user (freqserv=0)
csi_nru

## [1] 0.2401346

csi_ru  <- logis(1, gama)   ## csi parameter for regular user (freqserv=1)
csi_ru

## [1] 0.1231244
```

where the routine `logis()` returns the logistic transform componentwise of a standard matrix (binding the array Y of covariates with a vector of ones to include an intercept term, placed in the first column) multiplied with the parameter vector `param`. In conclusion, the feeling component is specified via:

$$logit(1-\xi_i) \quad = \quad -\gamma_0 - \gamma_1 \, freqserv_i, \qquad i = 1, \ldots, n.$$

In addition to a likelihood ratio test obtained by comparing the log-likelihoods of the two models, we remark that the reduction in BIC index from `BIC(cub_00)`$=7535.208$ to `BIC(cub_csi)`$=7431.773$ strongly supports the inclusion of the covariate `freqserv` in the model. For fitting purposes it is also possible to compute the X^2 statistic of Pearson by implementing:

```
pai<-coef(cub_csi)[1]
gama<-coef(cub_csi)[2:3]
data(univer)
pearson<-chi2cub(m=7, univer$officeho, W = univer$freqserv, pai, gama)

Degrees of freedom          ==>  df  = 9
Pearson Fitting measure     ==>  X^2 = 97.49491 (p-val.= 0 )
Deviance                    ==>  Dev = 101.0313 (p-val.= 0 )

str(pearson)
```

```
List of 3
 $ chi2: num 97.5
 $ df  : num 9
 $ dev : num 101
```

Furthermore, the call to `makeplot(cub_csi)` produces a plot comparing the two fitted probability distributions conditional on the values of the dichotomous covariate, as shown by Figure 4.3 (right panel).

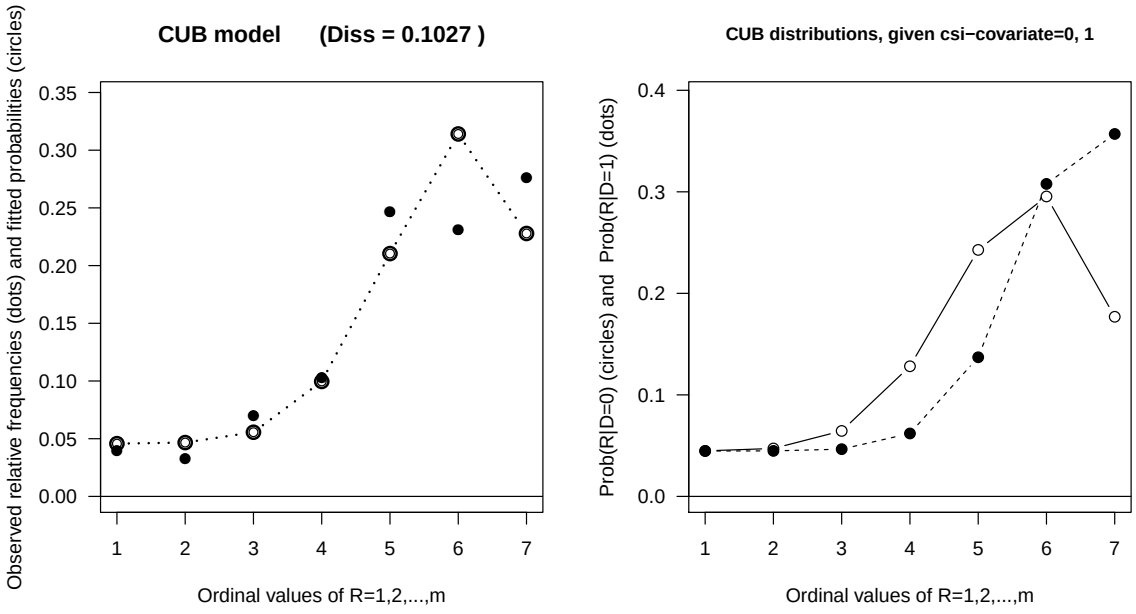


Figure 4.3: CUB without and with dichotomous covariate `freqserv` for feeling on `officeho`.

Note that the (normalized) dissimilarity index between observed and fitted probability distribution for a given fitted model without covariate is reported in the main title of the corresponding plot (see Section 4.5 for more details).

When covariates are specified for the feeling component, optimal preliminary estimates of the corresponding coefficients (Iannario 2008) are implemented via `inibestgama(m, ordinal, W)`, which is the option set to initialize the EM algorithm when calling `GEM(Formula~0|W|0,family="cub")`:

```
data(univer)
inicsicov<-inibestgama(m,univer$officeho,W=univer$freqserv)
inicsicov


gamma_0 -0.001596273
gamma_1 -0.247482482
```

We conclude this subsection by briefly presenting a case study in which a continuous covariate and a dichotomous one are jointly considered: in particular, we include the deviation from the mean of the logarithmic transform of `Age`, that is the covariate `lage`, and the covariate `gender` to explain both feeling and uncertainty.

```
data(univer)
age<-univer$age
lage<-log(age)-mean(log(age))          # Deviation from mean of logged Age
cub_pai_csi<-GEM(Formula(officeho~lage+gender|lage+freqserv|0),family="cub",data=univer)
summary(cub_pai_csi,correlation=TRUE,digits=3)

=============================================================================
=====>>> CUB  model    <<<=====   ML-estimates via E-M algorithm
=============================================================================
 m= 7  Sample size: n= 2179  Iterations= 23  Maxiter= 500
=============================================================================
Uncertainty
         Estimates StdErr Wald
constant     0.563  0.118 4.78
lage         1.240  0.613 2.02
gender       0.495  0.169 2.94
=============================================================================
Feeling
         Estimates StdErr   Wald
constant    -1.147 0.0404 -28.41
lage        -0.590 0.2410  -2.45
freqserv    -0.824 0.0851  -9.68
=============================================================================
Parameters Correlation matrix
           constant       lage     gender   constant       lage   freqserv
constant  1.0000000 -0.0506498 -0.6428539  0.1733713  0.0186039  0.1264982
lage     -0.0506498  1.0000000  0.0915196 -0.0064633  0.2750023 -0.0028018
gender   -0.6428539  0.0915196  1.0000000 -0.0158130 -0.0255944  0.0369170
constant  0.1733713 -0.0064633 -0.0158130  1.0000000 -0.0992397 -0.4153292
lage      0.0186039  0.2750023 -0.0255944 -0.0992397  1.0000000  0.0802712
freqserv  0.1264982 -0.0028018  0.0369170 -0.4153292  0.0802712  1.0000000
=============================================================================
Log-lik            = -3693.888
Mean Log-likelihood= -1.695
-----------------------------------------------------------------------------
AIC       = 7399.776
BIC       = 7433.895
ICOMP     = 7396.68
=============================================================================
Elapsed time= 1.49 seconds =====>>> Thu Feb 08 19:13:21 2018
=============================================================================
```

When covariates are included for both parameters as in the previous example, the estimates will be the corresponding coefficients of the logistic regression for the uncertainty and the feeling components:

```
coef(cub_pai_csi,digits=3)

##
## constant  0.563
## lage      1.240
## gender    0.495
## constant -1.147
## lage     -0.590
## freqserv -0.824
```

Then, the resulting CUB model can be summarized by:

$$\begin{cases} logit(1-\pi_i) &= -\beta_0 - \beta_1\, lage_i - \beta_2\, gender_i \\ logit(1-\xi_i) &= -\gamma_0 - \gamma_1\, lage_i - \gamma_2\, freqserv_i. \end{cases}$$

Since no plot is directly provided as output, the performance of the CUB model with significant covariates on feeling and uncertainty parameters may be summarized as in Figure 4.4, obtained by the following code:

```
data(univer)
age<-univer$age
average<-mean(log(age))
ageseq<-log(seq(17, 51, by = 0.1))-average
param<-coef(cub_pai_csi)
####################
paicov0<-logis(cbind(ageseq, 0), param[1:3])
paicov1<-logis(cbind(ageseq, 1), param[1:3])

csicov0<-logis(cbind(ageseq, 0), param[4:6])
csicov1<-logis(cbind(ageseq, 1), param[4:6])
####################
plot(1-paicov0, 1-csicov0, type = "n", col = "blue", cex = 1,
     xlim = c(0, 0.6), ylim = c(0.4, 0.9), font.main = 4, las = 1,
     main = "CUB models with covariates",
     xlab = expression(paste("Uncertainty     ", (1-pi))),
     ylab = expression(paste("Feeling      ", (1-xi))), cex.main = 0.9,
     cex.lab = 0.9)
lines(1-paicov1, 1-csicov1, lty = 1, lwd = 4, col = "red")
lines(1-paicov0, 1-csicov0, lty = 1, lwd = 4, col = "blue")
lines(1-paicov0, 1-csicov1, lty = 1, lwd = 4, col = "black")
lines(1-paicov1, 1-csicov0, lty = 1, lwd = 4, col = "green")
legend("bottomleft", legend = c("Man-User", "Man-Not User",
   "Woman-User", "Woman-Not User"), col = c("black", "blue", "red", "green"),
```

```
   lty = 1, text.col = c("black", "blue", "red", "green"), cex = 0.6)
text(0.1, 0.85, labels = "Young", offset = 0.3, cex = 0.8, font = 4)
text(0.5, 0.5, labels = "Elderly", offset = 0.3, cex = 0.8, font = 4)
```

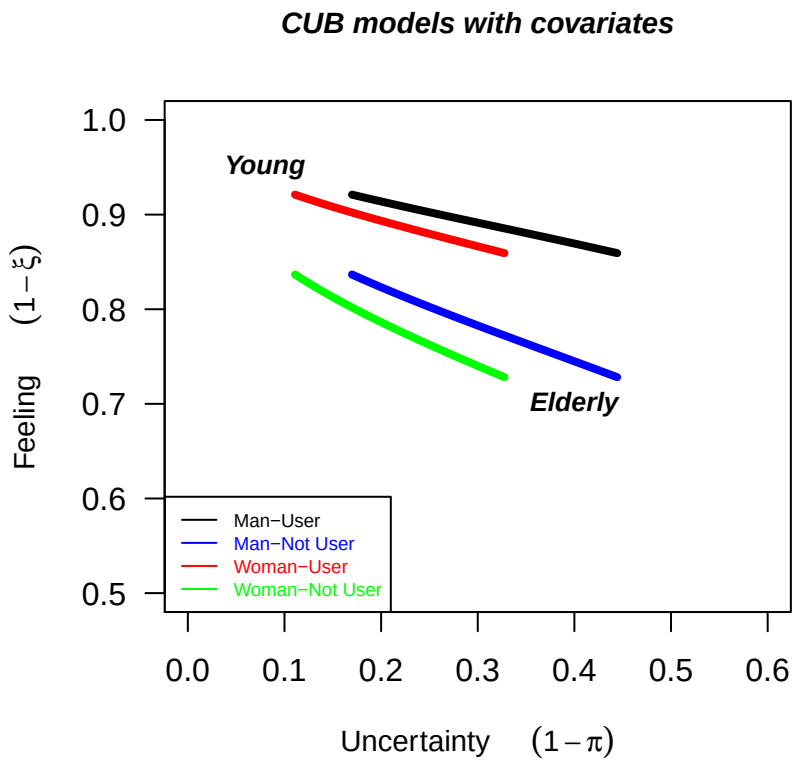


Figure 4.4: CUB models with covariates: logged `age` and `gender` for uncertainty and logged age and `freqserv` for feeling.

Summarizing, the sketch of analysis so pursued indicates that satisfaction increase with age whereas indecision decreases, and that men are more satisfied than women across all profiles.

4.2. Comparing IHG and CUBE models

As shown in Figure 4.1, CUBE models offer a wide flexibility in fitting data having different shapes and features. We now discuss how to perform an analysis on ordinal data with a CUBE model by considering the variable `willingn` within `data(univer)`.

```
starting<-c(0.5, 0.5, 0.1)
cubefit<-GEM(Formula(willingn~0|0|0),family="cube", starting = starting,
             maxiter = 100, toler = 1e-4,data=univer)
summary(cubefit,digits=7)

=========================================================================
```

```
=====>>> CUBE  model    <<<=====   ML-estimates via E-M algorithm
=========================================================================
 m= 7  Sample size: n= 2179  Iterations= 28  Maxiter= 100
=========================================================================
Uncertainty
    Estimates     StdErr     Wald
pai 0.8826023 0.01267281 69.64537
=========================================================================
Feeling
    Estimates      StdErr     Wald
csi 0.1248506 0.004427189 28.20087
=========================================================================
Overdispersion
    Estimates     StdErr     Wald
phi 0.0508419 0.01201249 4.232419
=========================================================================
Log-lik            = -2996.124
Mean Log-likelihood= -1.374999
Log-lik(UNIFORM)   = -4240.138
Log-lik(saturated) = -2988.635
Deviance           = 14.97685
-------------------------------------------------------------------------
AIC       = 5998.247
BIC       = 6015.307
ICOMP     = 5993.791
=========================================================================
Elapsed time= 0.03 seconds =====>>> Thu Feb 08 19:13:22 2018
=========================================================================
```

As shown with the current example, the user can set some extra options for the estimation procedure: the maximum number of iterations allowed for the optimization procedure (the default value for CUBE model is `maxiter = 1000`), the error tolerance to stop iterations (by changing the default option `toler = 1e-6`), and most importantly the argument `starting`, that is the vector of starting values to initialize the EM algorithm. The default option is based on the routine `inibestcube()`, a convenient choice of preliminary estimates for parameters in CUBE models. For CUBE models with covariates, only for feeling or for all the three parameters, initial estimates of parameters are implemented via `inibestcubecsi()` and `inibestcubecov()`, which are also set as default options when calling

```
GEM(Formula(ordinal~0|W|0),family="cube")
```

and

```
GEM(Formula(ordinal~Y|W|Z),family="cube")
```

respectively.

Given the shape of the distribution of the selected ordinal data (see Figure 4.5, left panel), the low level of uncertainty (`1-coef(cubefit)[1]`$= 0.1174$) and in particular, the extreme feeling (`1-coef(cubefit)[2]`$= 0.8752$) corresponding to the modal value at the last category $m = 7$, we check if the more parsimonious IHG model provides comparable goodness of fit. For such a model, the ML estimation procedure is initialized by considering as starting value the moment estimate of the preference parameter θ, computed via the function `iniihg(m,freq)`.

```
ihgfit<-GEM(Formula(willingn~0),family="ihg",data=univer)
summary(ihgfit,digits=7)

================================================================================
=====>>> IHG  model    <<<=====   ML-estimates via E-M algorithm
================================================================================
 m= 7  Sample size: n= 2179  Iterations= 1  Maxiter= 1
================================================================================
       Estimates      StdErr     Wald
theta 0.04756359 0.001414214 33.63254
================================================================================
Log-lik            = -3244.08
Mean Log-likelihood= -1.488793
Log-lik(UNIFORM)   = -4240.138
Log-lik(saturated) = -2988.635
Deviance           = 510.8901
--------------------------------------------------------------------------------
AIC       = 6490.16
BIC       = 6495.847
ICOMP     = 6488.16
================================================================================
Elapsed time= 0 seconds =====>>> Thu Feb 08 19:13:22 2018
================================================================================
```

In order to check if the fit improvement obtained with CUBE models justifies the inclusion of two additional parameters, that is, if the more parsimonious IHG model should be discarded in favor of CUBE model, we perform a likelihood ratio test:

```
llcube<-logLik(cubefit)
llihg<-logLik(ihgfit)
lrt<- -2*(llihg - llcube)    ### 495.9135
pv<- 1-pchisq(lrt, 2)        ### 0
```

It confirms that a consistently better fit is ensured by the CUBE model. Figure 4.5 displays the observed and fitted probability distributions for both models as returned by `makeplot(cubefit)` and `makeplot(ihgfit)`, respectively.

4.3. Models for shelter effect

As already mentioned in Section 2, the class of CUB mixture models includes a specific extension to fit the so-called *shelter effect*, arising in presence of an inflated category. We show how

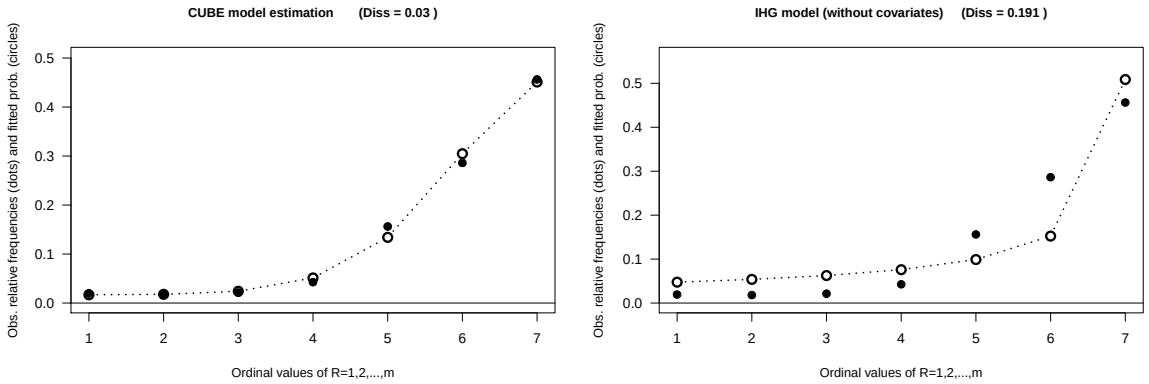


Figure 4.5: CUBE (left panel) and IHG (right panel) fit to the data.

to perform the analysis of a possible *shelter effect* on the data set `data(relgoods)`, a survey designed to assess importance of relational goods and involvement in leisure time activities: data were collected in December 2014 in the metropolitan area of Naples, Italy. Every participant was asked to measure his/her personal score for relational goods (for instance, time dedicated to friends and family) and his/her preferences towards leisure activities on a 10 point ordinal scale, ranging from 1 = "Not at all, nothing, never", to 10 = "Totally, extremely important, always". On the whole, respondents gave rating scores to 34 items. The sample is composed by 2459 interviews and 16 subjects' covariates (including gender, education, residence, marital status and age, among others). For these data some missing values are present.

Here, the implementation of the so-called *shelter effect* within CUB models is discussed. Let us consider the ordinal variable `Writing`, indicating the respondents' degree of engagement in writing as a preferred activity for leisure time.

One observes an excess in frequency corresponding to the first category, so we first fit an extended CUB model with *shelter effect* by running the following code (notice that both the available parameterizations (3) and (4) are reported on screen):

```
cub_she<-GEM(Formula(Writing~0|0|0),family="cub", shelter = 1,
             maxiter=500,toler=1e-3,data=relgoods)
summary(cub_she)

========================================================================
=====>>> CUB  model    <<<=====   ML-estimates via E-M algorithm
========================================================================
 m= 10  Sample size: n= 2449  Iterations= 77  Maxiter= 500
========================================================================
========================================================================
     Estimates     StdErr     Wald
pai1 0.1922306 0.01652574 11.63219
pai2 0.4902216 0.01907354 25.70166
csi  0.7518936 0.01433876 52.43784
```

```
=============================================================================
Alternative parameterization
        Estimates     StdErr     Wald
paistar 0.2816762 0.02422031 11.62975
csi     0.7518936 0.01433876 52.43784
delta   0.3175478 0.01121160 28.32315
=============================================================================
Log-lik            = -4887.843
Mean Log-likelihood= -1.995853
Log-lik(UNIFORM)   = -5639.031
Log-lik(saturated) = -4859.017
Deviance           = 57.65222
-----------------------------------------------------------------------------
AIC       = 9781.686
BIC       = 9799.096
ICOMP     = 9776.957
=============================================================================
Elapsed time= 0 seconds =====>>> Thu Feb 08 19:13:23 2018
=============================================================================
```

As shown by the printed output, the observed distribution is affected by a low weight for the shifted Binomial component (`pai1<-coef(cub_she)[1]`), and a moderately high weight for the Uniform component (`pai2<-coef(cub_she)[2]`): Figure 4.6 (left panel) shows the plot returned by running `makeplot(cub_she)`, which compares the observed frequencies and the fitted probabilities.

As a matter of fact the data set under consideration includes several instances of distributions characterized by a huge heterogeneity and an inflated category. As already explained in Section 2, such occurrence motivated the specification of CUSH models. For the variable `Writing` here considered, the fit of a CUSH model is implemented by the code:

```
cush<-GEM(Formula(Writing~0),family="cush",shelter = 1,data=relgoods)
summary(cush,digits=3)


=============================================================================
=====>>> CUSH  model    <<<=====   ML-estimates via E-M algorithm
=============================================================================
 m= 10  Sample size: n= 2449  Iterations= 1  Maxiter= 1
=============================================================================
      Estimates StdErr Wald
delta     0.313 0.0109 28.7
=============================================================================
Log-lik            = -4956.733
Mean Log-likelihood= -2.024
Log-lik(UNIFORM)   = -5639.031
Log-lik(saturated) = -4859.017
Deviance           = 195.432
```

```
------------------------------------------------------------------------
AIC       = 9915.466
BIC       = 9921.27
ICOMP     = 9913.466
========================================================================
Elapsed time= 0 seconds =====>>> Thu Feb 08 19:13:23 2018
========================================================================
```

Summarizing, CUB model with *shelter effect* provides a better fit in terms of log-likelihood and BIC indexes. As for other models within the GEM family, `makeplot(cush)` compares fitted and observed frequency distributions as displayed in Figure 4.6 (right panel).

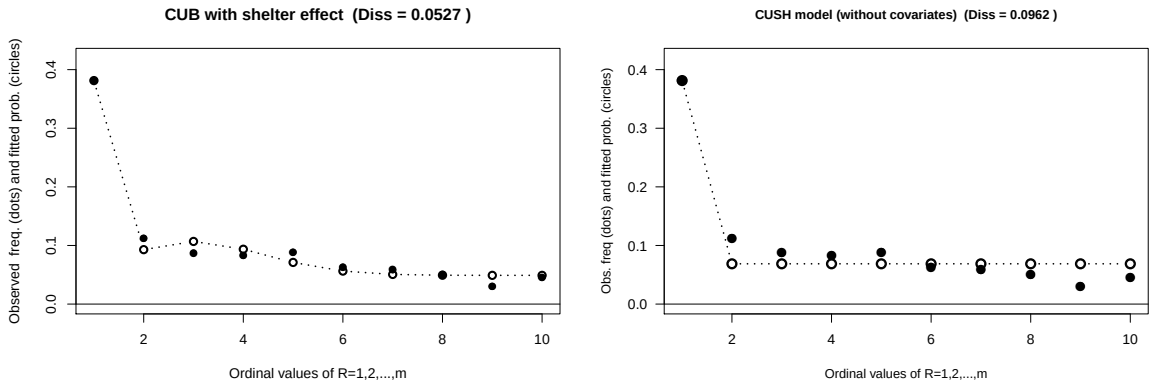


Figure 4.6: CUB with *shelter* (left panel) and CUSH (right panel) fit to the data.

The possibility of including covariates to explain the *shelter effect* in a CUSH model is presented in the next section along with the simulation routines offered by package **CUB**.

4.4. Simulation routines and experiments

Package **CUB** offers several facilities to perform simulation experiments involving GEM models. In order to generate n pseudo-random numbers from CUB, CUBE, CUB with *shelter*, CUSH and IHG distributions, respectively, the following functions are available:

```
simCUB   <- simcub(n, m, pai, csi)
simCUBE  <- simcube(n, m, pai, csi, phi)
simCUBshe <- simcubshe(n, m, pai, csi, delta, shelter)
simCUSH  <- simcush(n, m, delta, shelter)
simIHG   <- simihg(n, m, theta)
```

For instance, consider the following example, where the theoretical and fitted probability distributions of a CUB model are displayed for simulated data in Figure 4.7:

```
m<-9; n<-500
pai<-0.7; csi<-0.2
```

```
pr<-probcub00(m, pai, csi)
set.seed(123)
ordinal<-simcub(n, m, pai, csi)
cub<-GEM(Formula(ordinal~0|0|0),family="cub", maxiter = 50, toler = 1e-4)
pr_est<-fitted(cub)[,1]
plot(1:m, pr, type = "h", xlab = "Ordinal categories",
     ylab = "Probability", lwd = 3, ylim = c(0, 0.3))
vett<-1:m + 0.2
lines(vett, pr_est, type = "h", col = "blue", lwd = 3, lty = 2)
legend(1, 0.3, legend = c("Theoretical", "Fitted"), col = c("black", "blue"),
       lty = c(1, 2), lwd = 3, text.col = c("black", "blue"), bty = "n")
```

```
pai 0.66719
csi 0.18732
```

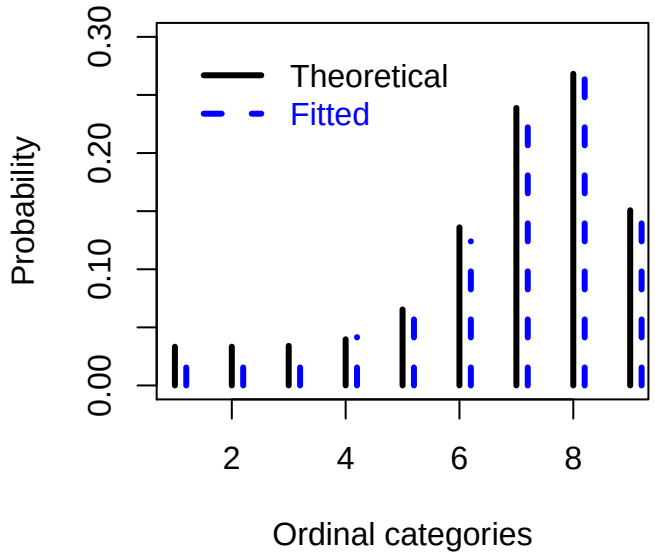


Figure 4.7: Fitted and theoretical CUB distributions on simulated data.

In order to simulate observations from a model with covariates, a two-step approach has to be implemented: one should first obtain the parameters $(\pi_i, \xi_i, \ldots)$ corresponding to the chosen respondents' profiles and then generate the sample data by using corresponding simulation routines with the given parameters. We outline how to implement such a procedure for a CUSH model with a dichotomous covariate for the *shelter effect*. Figure 4.8 displayes the two CUSH distributions conditional to the value of the covariate.

```
omega0<- -1.5
omega1<- -2
delta0<-as.numeric(logis(0, c(omega0, omega1)))  ## 0.1824255
delta1<-as.numeric(logis(1, c(omega0, omega1)))  ## 0.0293122
m<-9
n0<-700
n1<-1300
set.seed(1234)
ord0<-simcush(n0, m, delta0, shelter = s)
ord1<-simcush(n1, m, delta1, shelter = s)
ordinal<-c(ord0, ord1)
X<-c(rep(0, n0), rep(1, n1))
cushcov<-GEM(Formula(ordinal~X),family="cush",shelter = m)
coef(cushcov)
makeplot(cushcov)
```

```
constant -1.492551
X        -1.724813
```

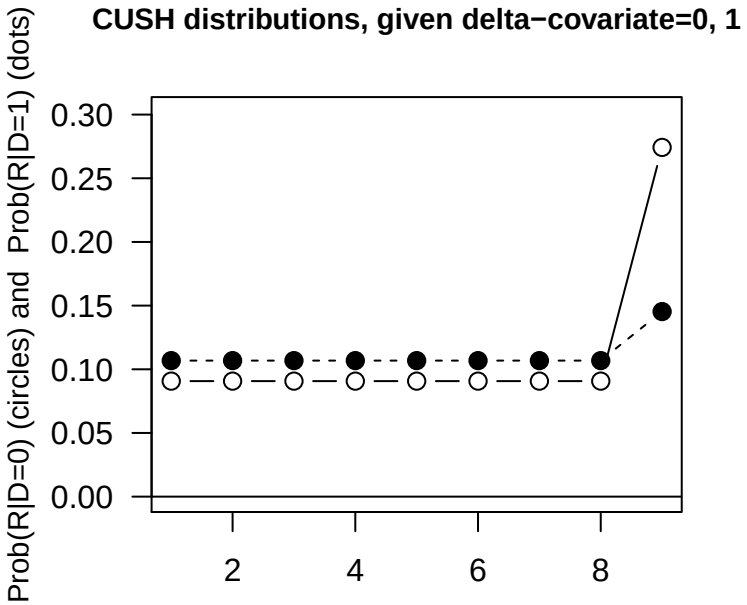


Figure 4.8: CUSH distributions with a dichotomous covariate.

4.5. Extra features of CUB package

The R package **CUB** provides a flexible framework for the analysis of ordinal data that covers some of the most common distributions related to GEM models. For a given discrete probability distribution, some additional features concern, for instance, the functions `gini()` and `laakso()`, for the normalized Gini (1912) and Laakso and Taagepera (1989) heterogeneity indexes, respectively, and the function `deltaprob(prob)` for the Mean Difference index (according to the de Finetti and Paciello (1930) formulation): Figure 4.9 shows an example of usage.

```
m<-7
pai<-0.4
csi<-0.2
prob<-probcub00(m, pai, csi)
Giniindex<-round(gini(prob), digits = 3)
Laaksoindex<-round(laakso(prob), digits = 3)
Delta<-round(deltaprob(prob), digits = 3)
plot(1:m, prob, type = "n", ylab = "", xlab = "", ylim = c(0, 0.4), las = 1)
lines(1:m, prob, type = "h", lwd = 3)
legend("topleft", xjust = 1, legend = c(paste("Gini       =", Giniindex),
                  paste("Laakso  =", Laaksoindex),
                  paste("Delta     =", Delta)), cex = 0.8)
```

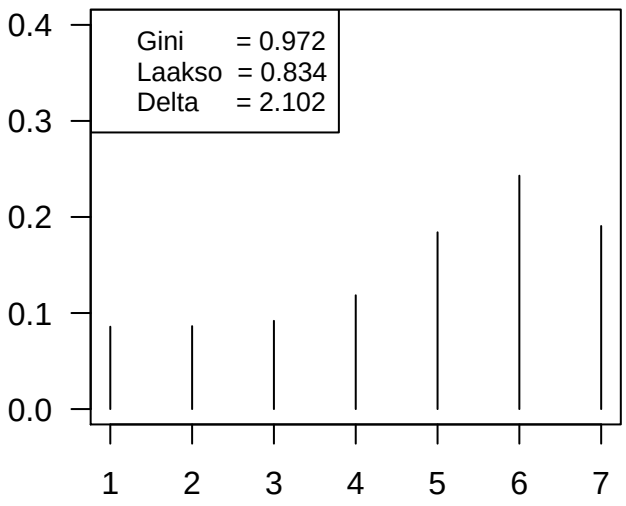


Figure 4.9: Heterogeneity and mutual variability indexes.

Another example concerns the simulation of the critical values for the normalized dissimilarity index between observed relative frequencies $(f_1, \ldots, f_m)$ and estimated (theoretical)

probabilities $(p_1(\hat{\boldsymbol{\theta}}), \ldots, p_m(\hat{\boldsymbol{\theta}}))$:

$$Diss = \frac{1}{2}\sum_{r=1}^{m} |f_r - p_r(\hat{\theta})|,$$

which is computed by the routine `dissim()`. This index evaluates the distance of the estimated model to a perfect fit. Due to its importance in model performance, this dissimilarity measure is computed when fitting CUB models and extensions within **CUB** package and displayed also in the plot title when it is returned. For this index we show how to run a simulation experiment for detecting the critical value for an assigned CUB model. The code concerns the distribution of a CUB model with $\pi = 0.3$ and $\xi = 0.8$. Figure 4.10 displays the results after `nsimul`$= 10000$ simulations of samples of sizes $n = 300$ over $m = 7$ categories.

```
pai<-0.3; csi<-0.8
nsimul<-10000
n<-300; m<-7
vectdiss<-rep(NA, nsimul)
for (j in 1:nsimul){
  ordinal<-simcub(n, m, pai, csi)
  mod<-GEM(Formula(ordinal~0|0|0),family="cub")
  theor<-fitted(mod)[,1]
  freq<-tabulate(ordinal, nbins = m)/n
  vectdiss[j]<-dissim(freq, theor)
}

sortvect<-sort(vectdiss)
alpha<-0.05
signif<-sortvect[(1-alpha)*nsimul] # empirical percentile 0.05
cr<-vectdiss[vectdiss>signif]       # critical region

effe<-density(vectdiss)
band<-effe$bw     # band width of the kernel plot
f<-function(x){   # compute kernel density (the default is Gaussian kernel)
  (1/(band*length(vectdiss)))*sum(dnorm((x-vectdiss)/band))
}
sortcr<-sort(cr)
sup<-numeric(length(cr))
for (j in 1:length(cr)){
  sup[j]=f(sortcr[j])    # compute kernel density values for critical values
}
title <- paste("Normalized Dissimilarity distribution, Critical level(0.05) = ",
               round(signif, 3))
plot(density(vectdiss), main = title, cex.main = 0.7, lwd = 3, xlab = "",
     cex.main = 0.7, las = 1)
polygon(c(signif, sortcr, sortcr[length(sortcr)], signif), c(0, sup, 0, 0),
        col = "gray")
abline(h=0)
```

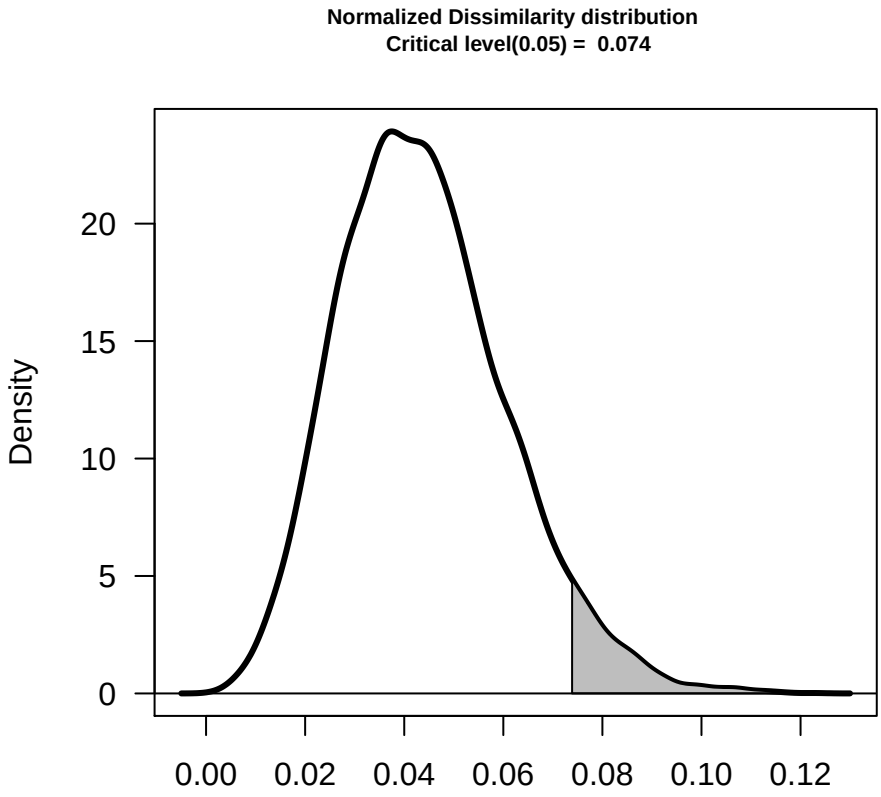


Figure 4.10: Distribution of the (normalized) dissimilarity index for simulated CUB model with $\pi = 0.3$ and $\xi = 0.8$.

We conclude the presentation of package **CUB** by underlying the effective graphical interpretation of parameters allowed by CUB models. When covariates are included to explain feeling and uncertainty, a *Scatter of Parameter Estimates* (SPE, for short) (Iannario and Piccolo 2014) reveals to be a convincing plotting device to identify different response patterns across subsets of respondents. A SPE simply performs the multiple representation of a CUB model as a point in the parameter space for each parameter vector (π_i, ξ_i) associated with each respondent. We show an example based on the `univer` dataset, by showing the CUB model for the `global` satisfaction conditioning both feeling and uncertainty on the evaluation expressed for `officeho`, and further specifying both components for varying age.

```
data(univer)
ordinal<-univer$global
lage<-log(univer$age)-mean(log(univer$age)) #Deviation from mean of logged Age
Y<-W<-cbind(univer$officeho,lage)
cub_pai_csi<-GEM(Formula(global~Y|W|0),family="cub",data=univer)
bet<-coef(cub_pai_csi)[1:3]
```

```
gama<-coef(cub_pai_csi)[4:6]
paivett<-logis(Y,bet)
csivett<-logis(W,gama)
n<-length(ordinal)
main<- "Scatter plot of estimated parameters"
vettcol<-symb<-rep(NA,n)
vettcol[univer$officeho<=3]<-"red"
vettcol[univer$officeho==4]<-"black"
vettcol[univer$officeho>=5]<-"blue"

symb[univer$officeho<=3]<-0
symb[univer$officeho==4]<-1
symb[univer$officeho>=5]<-2

plot(1-paivett,1-csivett,xlim=c(0,1),ylim=c(0,1),
     cex=0.8,pch=symb,col=vettcol,
     xlab=expression(1-pi),ylab=expression(1-xi),
     main=main,cex.main=1,font.main=4)
legend("topright",legend=c("Unsatisfied","Indifferent","Satisfied"),
       cex=0.7,text.col=c("red","black","blue"),pch=c(0,1,2),
       col=c("red","black","blue"))
```

5. Conclusions

Package **CUB** is under active development. Future plans include extra functions to fit CUBE models with further covariate specification, as well as higher flexibility for GeCUB models to possibly include covariates only for *shelter effect* or any pair of components. Implementation of Hierarchical CUB models and other extensions, including an adjusted version of CUBmodels to account for response styles and a multivariate model for repeated measurements are under scrutiny.

Package **CUB** has been implemented under R Version 3.2.5.

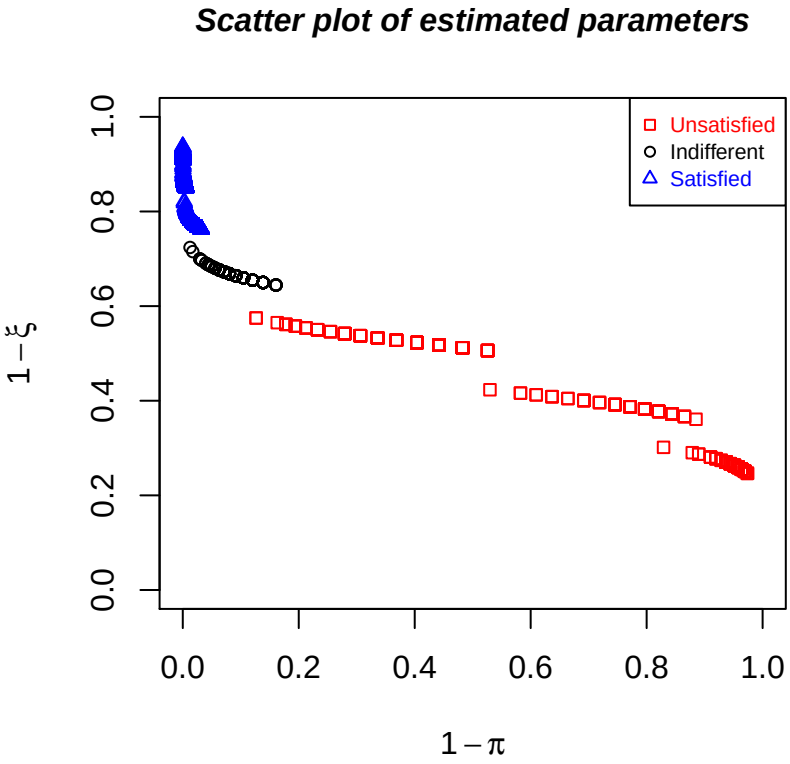
References

Agresti A. (2010). *Analysis of Ordinal Categorical Data*, 2^{nd} edition. J.Wiley & Sons, Hoboken.

Akaike H. (1974). A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, **19**(6), 716–723.

Andreis F., Ferrari P.A. (2013). On a copula model with CUBmargins. *Quaderni di Statistica*, **15**(1), 33–51.

Balirano G., Corduas M. (2008). Detecting Semiotically Expressed Humor in Diasporic TV Productions. *Humor*, **21**(3), 227–251.



Bodzogan H. (1990). On the information-based measure of covariance complexity and its application to the evaluation of multivariate linear models. *Communications in Statistics. Theory and Methods*, **19**(1), 221–278.

Capecchi S., Endrizzi I., Gasperi F. and Piccolo D. (2015). A multi-product approach for detecting subjects' and objects' covariates in consumer preferences. *British Food Journal*, **118**(3), 515–526.

Capecchi S., Iannario M. (2016). Gini heterogeneity index for detecting uncertainty in ordinal data surveys. *Metron*, **74**(2), 223–232.

Capecchi S., Piccolo D. (2014). Modelling the latent components of personal happiness, in: Perna and Sibillo (eds.): *Mathematical and Statistical Methods for Actuarial Sciences and Finance*, Springer Verlag, Berlin, pp.49–52.

Capecchi S., Piccolo D. (2016). Dealing with heterogeneity in ordinal responses. *Quality & Quantity*, **51**(5), 2375–2393.

Capecchi S., Piccolo D. and Simone R. (2017), An inflated model to account for large heterogeneity in ordinal data. In: *Data Science, Innovative Developments in Data Analysis and Clustering*, IFCS Proceedings, 205–217.

Christensen R.H.B. (2015). Package 'ordinal'. Regression Models for Ordinal Data. http://CRAN.R-project.org/package=ordinal.

Colombi R., Giordano S. (2016). A class of mixture models for multidimensional ordinal data, *Statistical Modelling: an International Journal*, **16**(4), 322–340.

Corduas M. (2015). Analyzing bivariate ordinal data with CUB margins. *Statistical Modelling: an International Journal*, **15**(5), 411–432.

Corduas M., Iannario M. and Piccolo D. (2009). A class of statistical models for evaluating services and performances, in: M. Bini *et al.* (eds.): *Statistical methods for the evaluation of educational services and quality of products*, Contribution to Statistics (pp.99–117). Berlin Heidelberg: Physica-Verlag, Springer.

D'Elia A. (2003). Modelling ranks using the Inverse Hypergeometric distribution. *Statistical Modelling: an International Journal*, **3**(1), 65–78.

D'Elia A. (2008). A statistical modelling approach for the analysis of TMD chronic pain data. *Statistical Methods in Medical Research*, **17**(4), 389–403.

D'Elia A., Piccolo D. (2005). A mixture model for preference data analysis. *Computational Statistics & Data Analysis*, **49**(3), 917–934.

de Finetti B., Paciello U. (1930), Calcolo della differenza media, *Metron*, **8**, 89–94.

Gambacorta R., Iannario M. (2013). Measuring job satisfaction with CUB models. *Labour*, **27**(2), 198–224.

Gambacorta R., Iannario M., Vaillant R. (2014). Design-based inference in a mixture model for ordinal variables for a two stage stratified design. *Australian and New Zealand Journal of Statistics*, **56**(2), 125–143.

Gini, C. (1912). *Variabilità e mutabilità*. Studi economico-giuridici, Facoltà di Giurisprudenza, Università di Cagliari, A, III, parte II.

Gottard A., Iannario M., Piccolo D. (2016). Varying uncertainty in CUB models. *Advances in Data Analysis and Classification*, **10**(2), 225–244.

Greene W.H., Hensher D.A. (2010). *Modeling Ordered Choices: A Primer*. Cambridge University Press.

Grilli L., Iannario M., Piccolo D., Rampichini C. (2014). Latent Class CUB Models. *Advances in Data Analysis and Classification*, **8**(1), 105–119.

Harrell F.E. Jr (2009). rms: Regression Modeling Strategies. R package version 2.1-0. http://CRAN.R-project.org/package=rms.

Iannario M. (2008). Selecting feeling covariates in rating surveys. *Rivista di Statistica Applicata*, **20**, 103–116.

Iannario M. (2010). On the identifiability of a mixture model for ordinal data. *Metron*, **LXVIII**(1), 87–94.

Iannario M. (2012a). Modelling *shelter* choices in a class of mixture models for ordinal responses. *Statistical Methods and Applications*, **21**(1), 1–22.

Iannario M. (2012b). CUBE models for interpreting ordered categorical data with overdispersion. *Quaderni di Statistica*, **14**(1), 137–140.

Iannario M. (2012c). Preliminary estimators for a mixture model of ordinal data. *Advances in Data Analysis and Classification*, **6**(3), 163–184.

Iannario M. (2012d). Hierarchical CUB models for ordinal variables. *Communications in Statistics. Theory and Methods*, **41**(16-17), 3110–3125.

Iannario M. (2014). Modelling Uncertainty and Overdispersion in Ordinal Data. *Communications in Statistics. Theory and Methods*, **43**(4), 771–786.

Iannario M., Piccolo D. (2014), Inference for CUB models: a program in R, *Statistica & Applicazioni*, **XII**, 177–204.

Iannario M. (2015). Detecting latent components in ordinal data with overdispersion by means of a mixture distribution. *Quality & Quantity*, **49**(3), 977–987.

Iannario M. (2016). Testing the overdispersion parameter in CUBE models. *Communications in Statistics: Simulation and Computation*, **45**(5), 1621–1635.

Iannario M., Piccolo D. (2010a). A New Statistical Model for the Analysis of Customer Satisfaction. *Quality Technology & Quantitative Management*, **7**(2), 149–168.

Iannario M., Piccolo D. (2010b). Statistical modelling of subjective survival probabilities. *GENUS*, **LXVI**(2), 17–42.

Iannario M., Piccolo D. (2016a). A comprehensive framework for regression models of ordinal Data. *Metron*, **74**(2), 233–252.

Iannario M., Piccolo D. (2016b). A generalized framework for modelling ordinal data. *Statistical Methods and Applications*, **25**, 163–189.

Iannario M., Manisera M., Piccolo D., Zuccolotto P. (2012). Sensory analysis in the food industry as a tool for marketing decisions. *Advances in Data Analysis and Classification*, **6**(4), 303–321.

Iannario M., Manisera M., Zuccolotto P. (2016). Treatment of "don't know" responses in the consumers' perceptions about sustainability in the agri-food sector. *Quality and Quantity*, DOI:10.1007/s11135-016-0438-7.

Iannario M., Monti A.C., Piccolo D. (2016). Robustness issues for CUB models. *TEST*, **25**(4), 731–750.

Iannario M., Monti A.C., Piccolo D., Ronchetti E. (2017). Robust inference for ordinal response models. *Electronic Journal of Statistics*, **11**(2), 3407–3445.

Laakso, M. and Taagepera, R. (1989). Effective number of parties: a measure with application to West Europe, *Comparative Political Studies*, **12**, 3–27.

Manisera M., Zuccolotto P. (2014a). Modeling "Don't know" responses in rating scales. *Pattern Recognition Letters*, **45**, 226–234.

Manisera M., Zuccolotto P. (2014b). Modeling rating data with Nonlinear CUB models. *Computational Statistics and Data Analysis*, **78**, 100–118.

Martin A. D., Quinn K. M., and Park J. H. (2009). MCMCpack: Markov Chain Monte Carlo in R. *Journal of Statistical Software*, **42**(9), 1–21.

McCullagh P. (1980). Regression models for ordinal data. *Journal of the Royal Statistical Society, Series B*, **42**(2), 109–142.

McLachlan G.J., Krishnan T. (1997). *The EM Algorithm and Extensions*. John Wiley & Sons, New York.

Piccolo D. (2003). On the moments of a mixture of uniform and shifted binomial random variables. *Quaderni di Statistica*, **5**(1), 85–104.

Piccolo D. (2006). Observed information matrix for MUB models. *Quaderni di Statistica*, **8**(1), 33–78.

Piccolo D. (2015). Inferential issues for CUBE models with covariates. *Communications in Statistics. Theory and Methods*, **44**(23), 771–786.

Piccolo D., D'Elia A. (2008). A new approach for modelling consumers' preferences. *Food Quality and Preference*, **19**(3), 247–259.

Ripley B.D., Venables, W.N. (2016). Package 'nnet'. http://CRAN.R-project.org/package=NNET.

Schwarz G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, **6**(2), 461–464.

Tutz G. (2012). *Regression for Categorical Data*. Cambridge University Press, Cambridge.

Tutz G., Schneider M., Iannario M. and Piccolo D. (2016). Mixture models for ordinal responses to account for uncertainty of choice. *Advances in Data Analysis and Classification*, **11**(2), 281–305.

Venables W.N., Ripley B.D. (2002). *Modern Applied Statistics with S*. Fourth edition. Springer.

Yee T.W. (2010). The VGAM Package for Categorical Data Analysis. *Journal of Statistical Software*, **32**(10), 1–34.

Zeileis A., Croissant Y. (2010). Extended Model Formulas in R: Multiple Parts and Multiple Responses. *Journal of Statistical Software*, **34**(1), 1–13.

Affiliation:

Maria Iannario, Domenico Piccolo, Rosaria Simone
Department of Political Sciences
University of Naples Federico II
80138 Naples, Italy

Telephone: +39/081-2537465
Fax: +39/081-2537466
E-mail: maria.iannario@unina.it,domenico.piccolo@unina.it,rosaria.simone@unina.it
URL: http://www.docenti.unina.it/MARIA.IANNARIO
URL: http://www.docenti.unina.it/DOMENICO.PICCOLO
URL: https://www.docenti.unina.it/ROSARIA.SIMONE