Multinomial Logit Models with Continuous and Discrete Individual Heterogeneity in R : The gmnl Package

Mauricio Sarrias
Cornell University

Ricardo A. Daziano
Cornell University

Abstract

This paper introduces the package **gmnl** in R for estimation of multinomial logit models with unobserved heterogeneity across individuals for cross-sectional and panel (longitudinal) data. Unobserved heterogeneity is modeled by allowing the parameters to vary randomly over individuals according to a continuous, discrete, or mixture distribution, which must be chosen a priori by the researcher. In particular, the models supported by **gmnl** are the multinomial or conditional logit, the mixed multinomial logit, the scale heterogeneity multinomial logit, the generalized multinomial logit, the latent class logit, and the mixed-mixed multinomial logit. These models are estimated using either the Maximum Likelihood Estimator or the Maximum Simulated Likelihood Estimator. This article describes and illustrates with real databases all functionalities of **gmnl**, including the derivation of individual conditional estimates of both the random parameters and willingness-to-pay measures.

Keywords: latent class, mixed multinomial logit, random parameters, preference heterogeneity, R.

1. Introduction

Modeling individual choices has been a very important avenue of research in diverse fields such as marketing, transportation, political science, and environmental, health, and urban economics. In all these areas the most widely used method to model choice among mutually exclusive alternatives has been the Conditional or Multinomial Logit model (MNL) (McFadden 1974), which belongs to the family of Random Utility Maximization (RUM) models. The main advantage of the MNL model has been its simplicity in terms of both estimation and interpretation of the resulting choice probabilities and elasticities. On the one hand, the MNL has a closed-form choice probability and a likelihood function that is globally concave. MNL estimation is thus straightforward using the Maximum Likelihood Estimator (MLE). On the other hand, it has been recognized that MNL not only imposes constant competition across alternatives – as a consequence of the independence of irrelevant alternatives (IIA) property – but also lacks the flexibility to allow for individual-specific preferences.

With the advent of more powerful computers and the improvement of simulation-aided inference in the last decades, researchers are no longer constrained to use models with closed-form solutions that may lead to unrealistic behavioral specifications. In fact, much of recent work focuses on extending MNL to allow for random-parameter models that accommodate unobserved preference heterogeneity.

The most popular MNL extension is the Mixed Logit Model (MIXL). MIXL allows parameters to vary randomly over individuals by assuming some continuous heterogeneity distribution a priori while keeping the MNL assumption that the error term is independent and identically distributed (i.i.d) extreme value type 1 (McFadden and Train 2000; Train 2009; Hensher and Greene 2003). MIXL is a very flexible model that can approximate any RUM model, and it does not exhibit the IIA property encountered in MNL. Furthermore, using the parametric heterogeneity distribution that describes how preferences vary in the population it is possible to derive conditional estimates of the parameters at the individual-level.

Latent Class (LC) discrete choice models offer an alternative to MIXL by replacing the continuous distribution assumption with a discrete distribution in which preference heterogeneity is captured by membership in distinct classes or segments (Boxall and Adamowicz 2002; Greene and Hensher 2003; Shen 2009). The standard LC specification is useful if the assumption of preference homogeneity holds within segments. In effect, all individuals in a given class have the same parameters (fixed parameters within a class), but the parameters vary across classes (heterogeneity across classes).

Bujosa, Riera, and Hicks (2010), and more recently Greene and Hensher (2013), have extended the LC model to allow for unobserved heterogeneity both within and across segments. The cross-group variation is modeled with the LC model, whereas the within-group variation is modeled as a continuous variation. An important characteristic of this model is that it nests both the MIXL and LC model in a double mixture specification. This model is also known as Mixed-Mixed Logit (MM-MNL) (Keane and Wasi 2013).

Other researchers have focused on MNL extensions that allow for a more flexible representation of heteroskedasticity. For example, Fiebig, Keane, Louviere, and Wasi (2010) proposed two new models, namely the Scale Heterogeneity (S-MNL) model and the Generalized Multinomial Logit (G-MNL) model. S-MNL extends the MNL by letting the scale of errors vary across individuals (via a parametric specification of heteroskedasticity), whereas the G-MNL nests the S-MNL, MIXL, and MNL models. For a discussion of confounding effects between scale and preference heterogeneity, see Hess and Rose (2012) and Hess and Stathopoulos (2013).

There exist different packages in R (R Core Team 2015) in order to estimate models with multinomial responses. Some packages that allow the estimation of Multinomial Logit model with fixed parameters are **mlogit** (Croissant 2012), **RSGHB** (Dumont, Keller, and Carpenter 2014), **mnlogit** (Hasan, Zhiyu, and Mahani 2015), the function `multinom` function from **nnet** package (Venables and Ripley 2002), **VGAM** (Yee 2010), and package **bayesm** (Rossi. 2012). The Multinomial Probit (MNP) model is fitted in **MNP** (Imai and Dyk 2005) and **mlogit** package. Models with random parameters are supported by **mlogit** and **RSGHB**. In terms of models with latent classes **bayesm**, **RSGHB**, **flexmix** (Leisch 2004), and **poLCA** (Linzer and Lewis 2011) offer alternative estimation procedures. Among all these packages, **mlogit** is probably the most user-friendly R package for the estimation of models with multinomial responses. Table 1 presents a more complete overview of the models supported by each package and the estimation procedure used to estimate the parameters.

The **gmnl** package (Sarrias and Daziano 2015) is intended to consolidate in a single R package the whole range of discrete choice models with random parameters for the use of researchers

Model	Package	Estimation Procedure
MNL	**mlogit**	Maximum likelihood
	mnlogit	Maximum likelihood
	RSGHB	Beyesian inference
	gmnl	Maximum likelihood
	`multinom` function (**nnet**)	Maximum likelihood
	VGAM	Maximum likelihood
	bayesm	Bayesian inference
MNP	**mlogit**	Maximum simulated likelihood
	MNP	Markov chain monte carlo
MIXL	**mlogit**	Maximum simulated likelihood
	gmnl	Maximum simulated likelihood
	RSGHB	Beyesian inference
G-MNL	**gmnl**	Maximum simulated likelihood
S-MNL	**gmnl**	Maximum simulated likelihood
LC-MNL	**bayesm**	Bayesian inference
	RSGHB	Bayesian inference
	flexmix	Expectation-Maximization
	poLCA	Expectation-Maximization
	gmnl	Maximum likelihood
MM-MNL	**gmnl**	Maximum simulated likelihood

Table 1: Packages available in R for models with multinomial response.

and practitioners. It shares similar functionalities with **mlogit** and **mnlogit** in terms of the formula interface. Furthermore, since **gmnl** is able to estimate G-MNL models, it also allows the user to estimate models in willingness-to-pay space with a minimal extra reformulation. Our package also provides the ability of constructing the conditional estimates for the individual parameters and willingness-to-pay. **gmnl** is available from the Comprehensive R Archive Network (CRAN) at `http://cran.r-project.org/package=gmnl`.

The paper is organized as follows: Section 2 presents a brief overview of the models supported by **gmnl**. Section 3 discusses the functionalities of the package. Section 4 concludes the paper.

2. Models

2.1. Mixed and latent class logit models

MIXL generalizes the MNL model by allowing the preference or taste parameters to be different for each individual (McFadden and Train 2000; Train 2009). MIXL is basically a random parameter logit model with continuous heterogeneity distributions. The random utility of person i for alternative j and for choice occasion t is:

$$U_{ijt} = \mathbf{x}_{ijt}^{\top}\boldsymbol{\beta}_i + \epsilon_{ijt} \quad i = 1, ..., N; j = 1, ..., J, t = 1, ..., T_i, \tag{1}$$

where $\mathbf{x}_{ijt}$ is a $K \times 1$ vector of observed alternative attributes; ϵ_{ijt} is the idiosyncratic error term or taste shock, and is i.i.d. extreme value type 1; the parameter vector $\boldsymbol{\beta}_i$ is unobserved for each i and is assumed to vary in the population following the continuous density $f(\boldsymbol{\beta}_i|\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ are the parameters of this distribution. This mixing distribution can in principle take any shape. For example, when assuming that the parameters are distributed multivariate normal, $\boldsymbol{\beta}_i \sim \text{MVN}(\boldsymbol{\beta}, \boldsymbol{\Sigma})$, the vector $\boldsymbol{\beta}_i$ can be re-written as:

$$\boldsymbol{\beta}_i = \boldsymbol{\beta} + \mathbf{L}\boldsymbol{\eta}_i,$$

where $\boldsymbol{\eta}_i \sim N(\mathbf{0}, \mathbf{I})$, and $\mathbf{L}$ is the lower-triangular Cholesky factor of $\boldsymbol{\Sigma}$ such that $\mathbf{L}\mathbf{L}^{\top} = \mathsf{VAR}(\boldsymbol{\beta}_i) = \boldsymbol{\Sigma}$. If the off-diagonal elements of $\mathbf{L}$ are zero, then the parameters are independently normally distributed. Observed heterogeneity (deterministic taste variations) can also be accommodated in the random parameters by including individual-specific covariates (see for example Greene 2012). Specifically, the vector of random coefficients is:

$$\boldsymbol{\beta}_i = \boldsymbol{\beta} + \boldsymbol{\Pi}\mathbf{z}_i + \mathbf{L}\boldsymbol{\eta}_i, \tag{2}$$

where $\mathbf{z}_i$ is a set of M characteristics of individual i that influence the mean of the preference parameters; and $\boldsymbol{\Pi}$ is a $K \times M$ is a matrix of additional parameters.

Unlike the MIXL model, LC uses a discrete mixing distribution, where individual i belongs to class q with probability w_{iq}, i.e.,:

$$\boldsymbol{\beta}_i = \boldsymbol{\beta}_q \;\text{ with probability }\; w_{iq} \;\text{ for }\; q = 1, ..., Q,$$

where $\sum_q w_{iq} = 1$ and $w_{iq} > 0$. The discrete mixing distribution (or class assignment probability) is unknown to the analyst. The most widely used formulation for w_{iq} is the semi-parametric Multinomial Logit format (Greene and Hensher 2003; Shen 2009):

$$w_{iq} = \frac{\exp\left(\mathbf{h}_i^{\top}\boldsymbol{\gamma}_q\right)}{\sum_{q=1}^{Q}\exp\left(\mathbf{h}_i^{\top}\boldsymbol{\gamma}_q\right)}; \;\; q = 1, ..., Q, \;\; \boldsymbol{\gamma}_1 = \mathbf{0},$$

where $\mathbf{h}_i$ denotes a set of socio-economic characteristics that determine assignment to classes. The parameters of the first class are normalized to zero for identification of the model. Note that one could omit any socio-economic covariate as a determinant of the class assignment probability. Under this scenario, the class probabilities simply become:

$$w_{iq} = \frac{\exp\left(\gamma_q\right)}{\sum_{q=1}^{Q}\exp\left(\gamma_q\right)};\;\; q = 1, ..., Q,\;\; \gamma_1 = 0,$$

where γ_q is a constant (Scarpa and Thiene 2005).

Let $y_{ijt} = 1$ if individual i chooses j on occasion t, and 0 otherwise. Then, the unconditional probabilities of the sequence of choices by individual i for MIXL and LC are respectively given by:

$$\begin{aligned}
P_i(\boldsymbol{\theta}) &= \int\left\{\prod_t^T\prod_j^J\left[\frac{\exp\left(\mathbf{x}_{ijt}^\top\boldsymbol{\beta}_i\right)}{\sum_{j=1}^J\exp\left(\mathbf{x}_{ijt}^\top\boldsymbol{\beta}_i\right)}\right]^{y_{ijt}}\right\}f(\boldsymbol{\beta}_i)d\boldsymbol{\beta}_i \\
P_i(\boldsymbol{\theta}) &= \sum_q^Q w_{iq}\left\{\prod_t^T\prod_j^J\left[\frac{\exp\left(\mathbf{x}_{ijt}^\top\boldsymbol{\beta}_q\right)}{\sum_{j=1}^J\exp\left(\mathbf{x}_{ijt}^\top\boldsymbol{\beta}_q\right)}\right]^{y_{ijt}}\right\}.
\end{aligned}$$

Both MIXL and LC are widely used in practice to accommodate preference heterogeneity across respondents. As discussed above, in the MIXL approach parameters are assumed to vary across the population according to some prespecified statistical distribution in a way that defines continuous segmentation of preferences. In the LC model a discrete number of separate classes or segments, each with different fixed parameters, recover preference heterogeneity. In addition to differentiation in terms of continuous versus discrete consumer segments, there exist further differences between MIXL and LC. For example, compared with the MIXL approach, the LC model has the advantage of being “relatively simple, reasonably plausible and statistically testable” (Shen 2009). In addition, because LC is a semiparametric specification that depends only on the prespecified number of classes, it avoids misspecification problems in the distribution of individual heterogeneity. In fact, the main disadvantage of MIXL is that the researcher has to choose the distribution of the random parameters a priori. Nevertheless, LC is less flexible than MIXL precisely because the parameters in each class are fixed. Another important difference between these two models is the estimation procedure. The MIXL requires the use of the maximum simulated likelihood estimator – which can be very costly in terms of computational time – but no simulation is required for LC.[1] **gmnl** implements the Maximum Likelihood Estimator for both LC and MIXL, using analytical expressions for the appropriate gradient.

To take advantage of the benefits of both models, recent empirical papers have derived a mixture of LC and MIXL. This double-mixture model is known as the ‘Mixed-Mixed’ Logit model (MM-MNL) (Keane and Wasi 2013).[2] Bujosa *et al.* (2010), and Greene and Hensher (2013) developed this MM-MNL model by extending the LC model to allow for random parameters within each class.

Consider the case where the heterogeneity distribution is generalized to a discrete mixture of

[1]For an empirical comparison between these two models, see for example Greene and Hensher (2003), Shen (2009) and Hess, Ben-Akiva, Gopinath, and Walker (2011).

[2]Train (2008) refers to this model as ‘discrete mixture of continuous distributions’, whereas Greene and Hensher (2013) label it ‘LC-MIXL’.

multivariate normal distributions. In this case we have:

$$\boldsymbol{\beta}_i \sim N(\boldsymbol{\beta}_q, \boldsymbol{\Sigma}_q) \;\text{ with probability }\; w_{iq} \;\text{ for }\; q = 1, ..., Q. \tag{3}$$

The appeal of using a Gaussian mixture for the heterogeneity distribution is that any continuous distribution can be approximated by a discrete mixture of normal distributions (Train 2008). Note that the MM-MNL with only one class is equivalent to the MIXL model. Furthermore, if $\boldsymbol{\Sigma}_q \to 0$ for all q, the model in Equation 3 becomes a LC-MNL model (Bujosa *et al.* 2010; Keane and Wasi 2013). Thus, MM-MNL nests both MIXL and LC.

The choice probabilities for the MM-MNL are given by:

$$P_i(\boldsymbol{\theta}) = \sum_q^Q w_{iq} \int \left\{ \prod_t^T \prod_j^J \left[\frac{\exp\left(\mathbf{x}_{ijt}^\top \boldsymbol{\beta}_i\right)}{\sum_{j=1}^J \exp\left(\mathbf{x}_{ijt}^\top \boldsymbol{\beta}_i\right)} \right]^{y_{ijt}} \right\} f(\boldsymbol{\beta}_i) d\boldsymbol{\beta}_i,$$

where $f(\boldsymbol{\beta}_i) = N(\boldsymbol{\beta}_q, \boldsymbol{\Sigma}_q)$. **gmnl** implements the maximum likelihood estimator for the MM-MNL parameters with the Monte-Carlo approximation of this choice probability and the analytical expression of the gradient.

2.2. Generalized Multinomial Logit model

Fiebig *et al.* (2010) proposed a general version of the MIXL model, which they called the G-MNL model, where the parameters vary across individuals according to:

$$\boldsymbol{\beta}_i = \sigma_i \boldsymbol{\beta} + \left[\gamma + \sigma_i(1-\gamma)\right] \mathbf{L}\boldsymbol{\eta}_i, \tag{4}$$

where σ_i is the individual-specific scale of the idiosyncratic error term, and γ is a scalar parameter that controls how the variance of residual taste heterogeneity $\mathbf{L}\boldsymbol{\eta}_i$ varies with scale. To better understand this specification, it is useful to note that differing sub-models arise when some structural parameters in the G-MNL model are constrained:

- G-MNL-I: If $\gamma = 1$, then $\boldsymbol{\beta}_i = \sigma_i\boldsymbol{\beta} + \mathbf{L}\boldsymbol{\eta}_i$. In this model, the residual taste heterogeneity is independent of the scaling of $\boldsymbol{\beta}$.

- G-MNL-II: If $\gamma = 0$, then $\boldsymbol{\beta}_i = \sigma_i(\boldsymbol{\beta} + \mathbf{L}\boldsymbol{\eta}_i)$. In this model, the residual taste heterogeneity is proportional to σ_i.

- S-MNL: If $\mathsf{VAR}(\boldsymbol{\eta}_i) = \mathbf{0}$, then $\boldsymbol{\beta}_i = \sigma_i\boldsymbol{\beta}$. As pointed out by Fiebig *et al.* (2010), this model is observationally equivalent to the particular type of heterogeneity in which the parameters increase or decrease proportionally across individuals by the scaling factor σ_i. S-MNL provides a more parsimonious representation of continuous heterogeneity than MIXL, because $\boldsymbol{\beta}\sigma_i$ is a simpler object than $\boldsymbol{\beta} + \mathbf{L}\boldsymbol{\eta}_i$ (Fiebig *et al.* 2010).

- MIXL: $\boldsymbol{\beta}_i = \boldsymbol{\beta} + \mathbf{L}\boldsymbol{\eta}_i$, if $\sigma_i = 1$

- MNL: $\boldsymbol{\beta}_i = \boldsymbol{\beta}$, if $\sigma_i = 1$ and $\mathsf{VAR}(\boldsymbol{\eta}_i) = \mathbf{0}$

Fiebig *et al.* (2010) note that some restrictions need to be considered to estimate the G-MNL model. First, the domain of σ_i should be the positive real line. A positive scale parameter is ensured by assuming that σ_i is distributed log-normal with standard deviation τ and mean $\bar{\sigma}$ Fiebig *et al.* (2010):

$$\sigma_i = \exp(\bar{\sigma} + \tau\upsilon_i),$$

where $\upsilon \sim N(0,1)$. Fiebig *et al.* (2010) also note that when τ is too large, numerical problems arise for extreme draws of υ_i. To avoid this numerical issue, the authors suggest to use a truncated normal distribution for υ_i with truncation at ± 2, so that $\upsilon \sim TN\left[-2,+2\right]$. Greene and Hensher (2010) found that constraining υ_i at -1.96 and $+1.96$ maintains the smoothness of the estimator. Specifically, the authors used $\upsilon_{ir} = \Phi^{-1}(0.025 + 0.95u_{ir})$, where u_{ir} is a draw from the standard uniform distribution. **gmnl** allows the user to choose between these two ways of drawing from υ_i, using the argument `typeR` (see Section 3.1).

Note that the parameters $\bar{\sigma}$, τ, and $\boldsymbol{\beta}$ are not separately identified. Fiebig *et al.* (2010) suggest that one can normalize the mean $\bar{\sigma}$ by setting:

$$\bar{\sigma} = -\log\left[\frac{1}{N}\sum_{i=1}^{N}\exp\left(\tau\upsilon_i\right)\right].$$

Another important issue in G-MNL is the domain of γ. Initially, Fiebig *et al.* (2010) imposed $\gamma \in [0,1]$. To constrain γ in this interval, the authors used the logistic transformation:

$$\gamma = \frac{\exp(\gamma^*)}{1+\exp(\gamma^*)},$$

and estimated γ^*. However, Keane and Wasi (2013) pointed out that both $\gamma < 0$ and $\gamma > 1$ still have meaningful behavioral interpretations. Thus, these authors estimate γ directly. **gmnl** allows to estimate γ using both procedures.

Finally, one can allow the mean of the scale to differ across individuals by including individual-specific characteristics. In this case the scale parameter can be written as:

$$\sigma_i = \exp(\bar{\sigma} + \boldsymbol{\delta}\mathbf{s}_i + \tau\upsilon_i),$$

where $\mathbf{s}_i$ is a vector of attributes of individual i.

In terms of computation, all models, except for the LC and the MNL model, are estimated in **gmnl** using the maximum simulated likelihood estimator (MSLE). For a complete derivation of the asymptotic properties of the MSLE and a more comprehensive review of how to implement this estimator, see Train (2009), Lee (1992), Gourieroux and Monfort (1997) or Hajivassiliou and Ruud (1986).

3. The gmnl package

3.1. A general overview of gmnl

All the models are estimated using the function `gmnl`, which has the following arguments.

```
gmnl(formula, data, subset, weights, na.action,
  model = c("mnl", "mixl", "smnl", "gmnl", "lc", "mm"),
  start = NULL, ranp = NULL, R = 40, Q = 2, haltons = NA,
  mvar = NULL, seed = 12345, correlation = FALSE,
  bound.err = 2, panel = FALSE,
  hgamma = c("direct", "indirect"),
  reflevel = NULL, init.tau = 0.1,
  init.gamma = 0.1, notscale = NULL,
  print.init = FALSE, gradient = TRUE,
  typeR = TRUE, ...)
```

A brief explanation of the main arguments is given below:

- `formula`: This is a symbolic description of the model to be estimated. The special formulae for Multinomial Logit models are given in Section 3.3.
- `data`: The data used for estimation, which must be of class `mlogit.data`. This is further explained in Section 3.2.
- `model`: A character string indicating which model will be estimated. The options are given in Table 2.

Options for `"model"`	Model
`"mnl"`	Multinomial Logit Model
`"mixl"`	Mixed Logit Model
`"smnl"`	Scaled Multinomial Logit Model
`"gmnl"`	Generalized Multinomial Logit Model
`"lc"`	Latent Class Multinomial Logit Model
`"mm"`	Mixed-Mixed Multinomial Logit Model

Table 2: Models supported by **gmnl**.

- `start`: A vector of starting values provided by the user.
- `ranp`: This is a named vector whose names are the random parameters and values the continuous distributions. This argument is valid only if the MIXL, G-MNL or MM-MIXL model is estimated. The distributions supported by **gmnl** are presented in Table 3.

  It is worth mentioning that given how the random parameters of the G-MNL model are constructed (see Equation 4), the distributions allowed when `model = "gmnl"` are the normal, uniform, and triangular. Similarly, when the model is estimated with correlated random parameters, only the normal distribution and its transformations —log-normal and truncated normal— are allowed.
- `R`: The number of draws used to simulate the probability if `ranp` is not `NULL`.
- `Q`: The number of classes for LC-MNL or MM-MNL model.

Shorthands	Distributions
`"n"`	Normal distribution
`"ln"`	Log-normal distribution
`"cn"`	Truncated (at zero) normal distribution
`"t"`	Triangular distribution
`"u"`	Uniform distribution
`"sb"`	Johnson S_b distribution

Table 3: Continuous distributions supported by **gmnl**.

- `haltons`: This argument is relevant if `ranp` is not `NULL`. If `haltons = NULL`, pseudo-random draws are used instead of Halton sequences. If `haltons = NA`, the first K primes are used to generate the Halton draws, where K is the number of random parameters, and 15 of the initial sequence of elements are dropped. Otherwise, `haltons` should be a list with elements `prime` and `drop`. For a further explanation of Halton draws see Train (2009).
- `mvar`: This argument is only valid if the model has observed heterogeneity in the mean of the random parameters. A more detailed discussion of this argument is presented in Section 3.5.
- `seed`: This is the seed for the random number generator if `haltons = NULL`.
- `correlation`: This argument is valid if `ranp` is not `NULL`. If `TRUE`, the correlation across random parameters is taken into account.
- `bound.err`: This argument is only relevant if the S-MNL or G-MNL model is estimated. It indicates at which values the draws for the scale parameter σ_i are truncated. By default `bound.err = 2`. Therefore, a truncated normal distribution with truncation at ± 2 for υ_i is used.
- `panel`: If `TRUE` a panel (longitudinal) data model is estimated.
- `hgamma`: A string character that indicates how to estimate the parameter γ in Equation 4. If `hgamma = "direct"`, then γ is estimated directly. If `hgamma = "indirect"`, then γ^* is estimated, where $\gamma = \exp(\gamma^*)/(1+\exp(\gamma^*))$. See Section 2.2.
- `init.tau`: Initial value for the τ parameter in Equation 4. The default is 0.1.
- `init.gamma`: Initial value for the γ parameter in Equation 4. The default is 0.1.
- `notscale`: This argument is relevant if the model being estimated is either S-MNL or G-MNL. It is a vector indicating which variables should not be scaled. See Section 3.4 for an illustration.
- `print.init`: If `TRUE`, then the initial values for the optimization procedure are displayed.
- `gradient`: If `TRUE`, then the analytical gradient is used for the optimization procedure. Otherwise, numerical approximation for the gradient is used.

- `typeR`: If `TRUE`, truncated normal draws are used for the scale parameter. In this case, the function `rtruncnorm` of **truncnorm** (Trautmann, Steuer, Mersmann, and Bornkamp 2014) is used. If `typeR = FALSE` the procedure suggested by Greene and Hensher (2010) is used. See Section 2.2.
- `...`: Further arguments passed to `maxLik` function of **maxLik** package (Henningsen and Toomet 2011).

3.2. Format of data

The function `mlogit.data` from **mlogit** is very useful to handle multinomial data formats. `gmnl` thus uses the same class of data for estimation. If the user forgets to set the data in the `mlogit.data` format, `gmnl` will give an error message and the estimation process will stop.

For illustration purposes, we use the Travel Mode data from the **AER** package (Kleiber and Zeileis 2008), which contains four transportation modes (`air`, `train`, `bus` and `car`), four alternative-specific variables (`wait`, `vcost`, `travel`, `gcost`), and two individual-specific variables (`income`, `size`).

```
data("TravelMode", package = "AER")
head(TravelMode)

##   individual  mode choice wait vcost travel gcost income size
## 1          1   air     no   69    59    100    70     35    1
## 2          1 train     no   34    31    372    71     35    1
## 3          1   bus     no   35    25    417    70     35    1
## 4          1   car    yes    0    10    180    30     35    1
## 5          2   air     no   64    58     68    68     30    2
## 6          2 train     no   44    31    354    84     30    2
```

This data base is in "long" format and can be transformed into the structure needed by `gmnl` using the `mlogit.data` in the following way:

```
library("mlogit")
TM <- mlogit.data(TravelMode, choice = "choice", shape = "long",
  alt.levels = c("air", "train", "bus", "car"))
```

The argument `choice` indicates the choice made by the individuals; `shape` specifies the original format of the data; and `alt.levels` is a character vector that contains the name of the alternatives. We show how to transform other kinds of data in the examples below. For a more complete treatment of the data using `mlogit.data` function see Croissant (2012).

3.3. Formula interface

The specification of Multinomial Logit models using **gmnl** is similar to that of **mlogit** and **mnlogit**. In particular, we use the R package **Formula** (Zeileis and Croissant 2010), which is able to handle multi-part formulae.

Consider the `TravelMode` data and suppose that we want to estimate a Multinomial Logit model where the variables `wait` and `vcost` are alternative-specific variables with a generic coefficient $\boldsymbol{\beta}$; `income` is an individual-specific variable with an alternative specific coefficient γ_j; and the variables `travel` and `gcost` are alternative-specific variables with an alternative specific coefficient $\boldsymbol{\delta}_j$. This is done using the following 3-part formula:

```
f1 <- choice ~ wait + vcost | income | travel + gcost
```

By default, the alternative-specific constants (ASC) for each alternative are included. They can be omitted by adding `+0` or `-1` in the second part of the formula. For example:

```
f2 <- choice ~ wait + vcost | income + 0 | travel + gcost
f2 <- choice ~ wait + vcost | income - 1 | travel + gcost
```

Some parts may be omitted when there is no ambiguity. For instance, a model with only individual specific variables can be specified as follows:

```
f3 <- choice ~ 0 | income + size | 0
f3 <- choice ~ 0 | income + size | 1
```

Similarly, a Conditional Logit model, that is, a model with alternative-specific variables with a generic coefficient $\boldsymbol{\beta}$, can be specified using either of the following `formula` objects:

```
f4 <- choice ~ wait + vcost |  0
f4 <- choice ~ wait + vcost |  0 | 0
f4 <- choice ~ wait + vcost | -1 | 0
```

For other models, such as the MIXL, S-MNL, LC-MNL and MM-MNL model, we require to use the fourth and fifth part of the `formula`. As explained in Section 2.1, **gmnl** allows incorporating observed heterogeneity in the mean of the random parameters. This can be achieved by including individual-specific characteristics (`income` and `size`) in the fourth part of the `formula`:

```
f5 <- choice ~ wait + vcost |  0 | 0 | income + size - 1
```

and then use the `mvar` argument to indicate how these two variables modify the mean of the random parameters. For a more complete example see Section 3.5.

The fifth part of the `formula` is reserved for either models with heterogeneity in the scale parameter or models with latent classes. For example, an S-MNL or G-MNL model where the scale varies across individuals by individual-specific characteristics can be specified as follows:

```
f6 <- choice ~ wait + vcost | 1 | 0 | 0 | income + size - 1
```

The same formulation can be used if a model with latent classes is estimated and both `income` and `size` determine the class assignment.

3.4. Estimating S-MNL models

In this example, we estimate an S-MNL model using the `TravelMode` data where the ASCs are fixed and not scaled. Fiebig *et al.* (2010) found that in a model where all attributes are scaled – including the ASCs – the estimates often show a explosive behavior and the model actually produces a worse fit. The basic syntax for estimation is the following:

```
library("gmnl")
smnl.nh <- gmnl(choice ~ wait + vcost + travel + gcost| 1,
  data = TM,
  model = "smnl",
  R = 30,
  notscale = c(1, 1, 1, rep(0, 4)))

##
## The following variables are not scaled:
## [1] "train:(intercept)" "bus:(intercept)"   "car:(intercept)"
## Estimating SMNL model
```

The component `| 1` in the `formula` means that the model is fitted using ASCs for the $J-1$ alternatives. The main argument in the model is `model = "smnl"`, which indicates to the function that the user wants to estimate the S-MNL model (without random parameters). `R = 30` indicates that 30 draws are used to simulate the probabilities. Another important argument in this example is `notscale`. This is a vector that indicates which variables will not be scaled ($1 =$ not scaled and $0 =$ scaled). Since the ASCs are always the first variables entering in the model (if they are specified using `| 1` in the second part of `formula`) and only $J-1=3$ ASCs are created, `notscale = c(1, 1, 1, rep(0, 4))` implies that the constants will not be scaled.

```
summary(smnl.nh)

##
## Model estimated on: Thu Jun 04 13:14:54 2015
##
## Call:
## gmnl(formula = choice ~ wait + vcost + travel + gcost | 1, data = TM,
##     model = "smnl", R = 30, notscale = c(1, 1, 1, rep(0, 4)),
##     method = "bfgs")
##
## Frequencies of categories:
##
##   air train   bus   car
## 0.276 0.300 0.143 0.281
##
## The estimation took: 0h:0m:4s
##
## Coefficients:
```

```
##                     Estimate Std. Error z-value Pr(>|z|)
## train:(intercept) -1.18012    0.58094   -2.03  0.04222 *
## bus:(intercept)   -1.92725    0.70229   -2.74  0.00607 **
## car:(intercept)   -7.07657    1.30568   -5.42  6.0e-08 ***
## wait              -0.13366    0.02070   -6.46  1.1e-10 ***
## vcost             -0.11741    0.03166   -3.71  0.00021 ***
## travel            -0.01721    0.00391   -4.40  1.1e-05 ***
## gcost              0.09229    0.02591    3.56  0.00037 ***
## tau                0.43016    0.13256    3.24  0.00118 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
## Log Likelihood: -180
## Number of observations: 210
## Number of iterations: 63
## Exit of MLE: successful convergence
## Simulation based on 30 draws
```

The results report the point estimates for each variable and τ, which represents the standard deviation of σ_i. The output also gives useful estimation information. The model is estimated using the BFGS procedure. Other optimization procedures such as the BHHH and Newton Raphson (NR) can be called using the argument `method` passed to the `maxLik` function.[3]

Another important point is that the number of observations reported by `gmnl` corresponds to N/J if cross-sectional data is used, or $N \times T/J$ if panel data (repeated choice situations) is used. Finally, it is always important to check all the details in the estimation output. In our example, the output informs us that the convergence was achieved successfully.

In the next example, we allow the scale to differ across individuals according to their income. Basically, we assume that:

$$\sigma_i = \exp\left(\bar{\sigma} + \delta_{\texttt{income}}\texttt{income}_i + \tau \upsilon_i\right).$$

The syntax is very similar to our previous example, with minor changes in the `formula` argument:

```
smnl.het <- gmnl(choice ~ wait + vcost + travel + gcost| 1 |
  0 | 0 | income - 1,
  data = TM,
  model = "smnl",
  R = 30,
  notscale = c(1, 1, 1, 0, 0, 0, 0))

##
## The following variables are not scaled:
```

[3]For more information about the arguments of this function type `help(maxLik)`.

```
## [1] "train:(intercept)" "bus:(intercept)"   "car:(intercept)"
## Estimating SMNL model
```

The fifth part of the `formula` is reserved for individual-specific variables that affect scale. In this example, we specify that the variable `income` and no constant are included in σ_i.

```
summary(smnl.het)

##
## Model estimated on: Thu Jun 04 13:15:00 2015
##
## Call:
## gmnl(formula = choice ~ wait + vcost + travel + gcost | 1 | 0 |
##     0 | income - 1, data = TM, model = "smnl", R = 30, notscale = c(1,
##     1, 1, 0, 0, 0, 0), method = "bfgs")
##
## Frequencies of categories:
##
##   air train   bus   car
## 0.276 0.300 0.143 0.281
##
## The estimation took: 0h:0m:5s
##
## Coefficients:
##                   Estimate Std. Error z-value Pr(>|z|)
## train:(intercept) -0.84832    0.61257   -1.38  0.16610
## bus:(intercept)   -1.50344    0.74401   -2.02  0.04331 *
## car:(intercept)   -6.69212    1.32771   -5.04  4.6e-07 ***
## wait              -0.11156    0.02048   -5.45  5.1e-08 ***
## vcost             -0.09060    0.02824   -3.21  0.00133 **
## travel            -0.01428    0.00344   -4.15  3.4e-05 ***
## gcost              0.07461    0.02241    3.33  0.00087 ***
## tau                0.46669    0.14599    3.20  0.00139 **
## het.income         0.00583    0.00339    1.72  0.08521 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
## Log Likelihood: -178
## Number of observations: 210
## Number of iterations: 75
## Exit of MLE: successful convergence
## Simulation based on 30 draws
```

The results are very similar to those of the previous example. All the parameters for the variables that enter in the scale are preceded by the string `het`. Thus, the coefficient `het.income` corresponds to $\delta_{\texttt{income}}$.

Suppose now that we want to test the null hypothesis $H_0 : \delta_{\texttt{income}} = 0$. This test can be performed using the function `waldtest` or `lrtest` from the package **lmtest** (Zeileis and Hothorn 2002):

```
library("lmtest")
```

```
waldtest(smnl.nh, smnl.het)

## Wald test
##
## Model 1: choice ~ wait + vcost + travel + gcost | 1
## Model 2: choice ~ wait + vcost + travel + gcost | 1 | 0 | 0 | income -
##     1
##   Res.Df Df Chisq Pr(>Chisq)
## 1    202
## 2    201  1  2.96      0.085 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

lrtest(smnl.nh, smnl.het)

## Likelihood ratio test
##
## Model 1: choice ~ wait + vcost + travel + gcost | 1
## Model 2: choice ~ wait + vcost + travel + gcost | 1 | 0 | 0 | income -
##     1
##   #Df LogLik Df Chisq Pr(>Chisq)
## 1   8   -180
## 2   9   -178  1  3.81      0.051 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.5. Estimating MIXL models

In the following examples we show how to estimate MIXL models using `gmnl`. The package **mlogit** is very efficient in estimating MIXL models. However, one advantage of using `gmnl` is the inclusion of individual-specific variables to explain the mean of the random parameters (see Equation 2). Other important expansions include the possibility of producing point and interval estimates at the individual level, and the consideration of Johnson S_b heterogeneity distributions.

If we assume that the coefficients of `travel` and `wait` vary across individuals according to:

$$\begin{aligned}\beta_{\texttt{travel},i} &= \beta_1 + \pi_{11}\texttt{income} + \pi_{12}\texttt{size} + \sigma_1\eta_{1i} \\ \beta_{\texttt{wait},i} &= \beta_2 + \pi_{21}\texttt{income} + \sigma_2\eta_{2i},\end{aligned}$$

where η_{1i} is triangular and $\eta_{2i} \sim N(0,1)$, the corresponding MIXL model is estimated by typing:

```
mixl.hier <- gmnl(choice ~ vcost + gcost + travel + wait | 1 |
  0 | income + size - 1,
  data = TM,
  model = "mixl",
  ranp  = c(travel = "t", wait = "n"),
  mvar = list(travel = c("income","size"),
              wait = c("income")),
  R = 50,
  haltons =  list("primes"= c(2, 17),
                  "drop" = rep(19, 2)))

## Estimating MIXL model
```

The argument `model = "mixl"` indicates that the MIXL model will be estimated. The distribution of the random coefficients are specified by the argument `ranp` (See Table 3 for shorthands of other continuous distributions allowed by **gmnl**). Note also that the fourth part of the `formula` is reserved for all the variables that enter the mean of the random parameters. The argument `mvar` indicates which variables enter each specific random parameter. For example `travel = c("income","size")` indicates that the mean of the `travel` coefficient varies according to `income` and `size`. Finally, `haltons` indicates the prime numbers used for the Halton draws and how many elements to drop for each random parameter.

```
summary(mixl.hier)

##
## Model estimated on: Thu Jun 04 13:15:40 2015
##
## Call:
## gmnl(formula = choice ~ vcost + gcost + travel + wait | 1 | 0 |
##     income + size - 1, data = TM, model = "mixl", ranp = c(travel = "t",
##     wait = "n"), R = 50, haltons = list(primes = c(2, 17), drop = rep(19,
##     2)), mvar = list(travel = c("income", "size"), wait = c("income")),
##     method = "bfgs")
##
## Frequencies of categories:
##
##   air train   bus   car
## 0.276 0.300 0.143 0.281
##
## The estimation took: 0h:0m:40s
##
## Coefficients:
##                      Estimate Std. Error z-value Pr(>|z|)
```

```
## train:(intercept) -3.15e-01   1.04e+00   -0.30  0.76224
## bus:(intercept)   -1.10e+00   1.10e+00   -1.01  0.31471
## car:(intercept)   -7.97e+00   2.06e+00   -3.86  0.00011 ***
## vcost             -5.10e-02   4.58e-02   -1.11  0.26566
## gcost              2.82e-02   4.51e-02    0.62  0.53293
## travel            -9.67e-03   5.21e-03   -1.86  0.06357 .
## wait              -1.33e-01   3.93e-02   -3.39  0.00070 ***
## travel.income     -1.28e-04   5.63e-05   -2.27  0.02302 *
## travel.size        2.22e-03   1.27e-03    1.74  0.08140 .
## wait.income       -1.12e-03   5.53e-04   -2.02  0.04385 *
## sd.travel          2.41e-03   6.07e-03    0.40  0.69098
## sd.wait            6.95e-02   2.77e-02    2.51  0.01216 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
## Log Likelihood: -164
## Number of observations: 210
## Number of iterations: 163
## Exit of MLE: successful convergence
## Simulation based on 50 draws
```

The output shows the estimates in the following order: fixed parameters, mean of the random parameters, effect of the variables that affect the mean of the random parameters, and finally the standard deviation/spread of the random parameters. Note that `travel.income` corresponds to π_{11}, `travel.size` corresponds to π_{12}, and `wait.income` corresponds to π_{21}.

We now estimate a correlated random parameter model using the `Electricity` data from the **mlogit** package, which is a panel dataset. Given time compilation restrictions, in this example we will use just a subsample of this database (`subset = 1:3000`). The user may want to use the whole sample to reproduce this case study.

```
data("Electricity", package = "mlogit")
Electr <- mlogit.data(Electricity, id.var = "id", choice = "choice",
  varying = 3:26, shape = "wide", sep = "")
```

In this example, two arguments are especially relevant in the `gmnl` function. First, `panel = TRUE` indicates that the data is a panel. When using panel data the user needs to specify a variable in the `id.var` argument of the `mlogit.data` function. Second, to estimate correlated random parameters `correlation = TRUE` needs to be indicated in the `gmnl` function. The syntax is the following:

```
Elec.cor <- gmnl(choice ~ pf + cl + loc + wk + tod + seas| 0,
  data = Electr,
  subset = 1:3000,
  model = 'mixl',
  R = 50,
```

```
  panel = TRUE,
  ranp = c(cl = "n", loc = "n", wk = "n",
           tod = "n", seas = "n"),
  correlation =  TRUE)

## Estimating MIXL model

summary(Elec.cor)

##
## Model estimated on: Thu Jun 04 13:15:54 2015
##
## Call:
## gmnl(formula = choice ~ pf + cl + loc + wk + tod + seas | 0,
##     data = Electr, subset = 1:3000, model = "mixl", ranp = c(cl = "n",
##         loc = "n", wk = "n", tod = "n", seas = "n"), R = 50,
##     correlation = TRUE, panel = TRUE, method = "bfgs")
##
## Frequencies of categories:
##
##     1     2     3     4
## 0.215 0.303 0.217 0.265
##
## The estimation took: 0h:0m:15s
##
## Coefficients:
##              Estimate Std. Error z-value Pr(>|z|)
## pf            -0.8702     0.0786  -11.07  < 2e-16 ***
## cl            -0.1765     0.0430   -4.11  4.0e-05 ***
## loc            2.3822     0.3053    7.80  6.0e-15 ***
## wk             1.9447     0.2493    7.80  6.2e-15 ***
## tod           -8.5026     0.7423  -11.45  < 2e-16 ***
## seas          -8.6456     0.7803  -11.08  < 2e-16 ***
## sd.cl.cl       0.3919     0.0420    9.33  < 2e-16 ***
## sd.cl.loc      0.4921     0.1983    2.48  0.01311 *
## sd.cl.wk       0.5514     0.2131    2.59  0.00966 **
## sd.cl.tod     -0.9834     0.2802   -3.51  0.00045 ***
## sd.cl.seas    -0.1470     0.2297   -0.64  0.52206
## sd.loc.loc     2.5925     0.4226    6.14  8.5e-10 ***
## sd.loc.wk      1.9311     0.3610    5.35  8.8e-08 ***
## sd.loc.tod     1.0198     0.5651    1.80  0.07114 .
## sd.loc.seas    0.0941     0.4579    0.21  0.83723
## sd.wk.wk      -0.3330     0.2212   -1.51  0.13226
## sd.wk.tod      1.9341     0.3208    6.03  1.7e-09 ***
## sd.wk.seas     0.7349     0.3030    2.43  0.01529 *
## sd.tod.tod     2.0635     0.3301    6.25  4.1e-10 ***
```

```
## sd.tod.seas    1.1689     0.2539    4.60  4.2e-06 ***
## sd.seas.seas   1.7034     0.2533    6.72  1.8e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
## Log Likelihood: -692
## Number of observations: 750
## Number of iterations: 97
## Exit of MLE: successful convergence
## Simulation based on 50 draws
```

The estimates from `sd.cl.cl` to `sd.seas.seas` are the elements of the lower triangular matrix $\mathbf{L}$. If the user is interested in the standard errors of the variance-covariance matrix of the random parameters $\mathbf{L}\mathbf{L}^\top = \boldsymbol{\Sigma}$ or the standard deviations, the `S3` function `vcov` can be used for finding these elements. The syntax for both cases is the following:[4]

```
vcov(Elec.cor, what = 'ranp', type = 'cov', se = 'true')

##
## Elements of the variance-covariance matrix
##
##              Estimate Std. Error t-value Pr(>|t|)
## v.cl.cl        0.1536     0.0329    4.67  3.1e-06 ***
## v.cl.loc       0.1928     0.0816    2.36   0.0181 *
## v.cl.wk        0.2161     0.0917    2.36   0.0185 *
## v.cl.tod      -0.3854     0.1290   -2.99   0.0028 **
## v.cl.seas     -0.0576     0.0906   -0.64   0.5249
## v.loc.loc      6.9630     2.2065    3.16   0.0016 **
## v.loc.wk       5.2776     1.6637    3.17   0.0015 **
## v.loc.tod      2.1599     1.3323    1.62   0.1050
## v.loc.seas     0.1715     1.1222    0.15   0.8785
## v.wk.wk        4.1440     1.3293    3.12   0.0018 **
## v.wk.tod       0.7832     0.8530    0.92   0.3585
## v.wk.seas     -0.1441     0.7525   -0.19   0.8481
## v.tod.tod     10.0058     3.4763    2.88   0.0040 **
## v.tod.seas     4.0739     1.5217    2.68   0.0074 **
## v.seas.seas    4.8384     1.1851    4.08  4.5e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

vcov(Elec.cor, what = 'ranp', type = 'sd', se = 'true')

##
```

[4] To compute the standard errors, **gmnl** uses the `deltamethod` function from the **msm** package (Jackson 2011).

```
## Standard deviations of the random parameters
##
##      Estimate Std. Error t-value Pr(>|t|)
## cl      0.392      0.042    9.33  < 2e-16 ***
## loc     2.639      0.418    6.31  2.8e-10 ***
## wk      2.036      0.327    6.23  4.5e-10 ***
## tod     3.163      0.549    5.76  8.6e-09 ***
## seas    2.200      0.269    8.17  2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The correlation matrix of the random parameters can be recovered using the following syntax:

```
vcov(Elec.cor,  what = 'ranp', type = 'cor')

##            cl    loc      wk    tod    seas
## cl    1.0000 0.1865  0.2708 -0.311 -0.0668
## loc   0.1865 1.0000  0.9825  0.259  0.0295
## wk    0.2708 0.9825  1.0000  0.122 -0.0322
## tod  -0.3109 0.2588  0.1216  1.000  0.5855
## seas -0.0668 0.0295 -0.0322  0.586  1.0000
```

3.6. Estimating G-MNL models

In the following examples we show how to estimate G-MNL models in **gmnl**. We will assume that the ASCs are random. Using the `formula` to create the ASCs produces problems in the `ranp` argument due to the way the constants are labeled. So, we first create the ASCs *by hand*:

```
Electr$asc2   <- as.numeric(Electr$alt == 2)
Electr$asc3   <- as.numeric(Electr$alt == 3)
Electr$asc4   <- as.numeric(Electr$alt == 4)
```

The G-MNL model is estimated using `model = "gmnl"`:

```
Elec.gmnl <- gmnl(choice ~ pf + cl + loc + wk + tod + seas +
  asc2 + asc3 + asc4 | 0,
  data = Electr,
  subset = 1:3000,
  model = 'gmnl',
  R = 50,
  panel = TRUE,
  notscale = c(rep(0, 6), 1, 1, 1),
  ranp = c(cl = "n", loc = "n", wk = "n",
          tod = "n", seas = "n",
          asc2 = "n", asc3 = "n", asc4 = "n"))
```

```
##
## The following variables are not scaled:
## [1] "asc2" "asc3" "asc4"
## Estimating GMNL model

summary(Elec.gmnl)

##
## Model estimated on: Thu Jun 04 13:16:22 2015
##
## Call:
## gmnl(formula = choice ~ pf + cl + loc + wk + tod + seas + asc2 +
##     asc3 + asc4 | 0, data = Electr, subset = 1:3000, model = "gmnl",
##     ranp = c(cl = "n", loc = "n", wk = "n", tod = "n", seas = "n",
##         asc2 = "n", asc3 = "n", asc4 = "n"), R = 50, panel = TRUE,
##     notscale = c(rep(0, 6), 1, 1, 1), method = "bfgs")
##
## Frequencies of categories:
##
##     1     2     3     4
## 0.215 0.303 0.217 0.265
##
## The estimation took: 0h:0m:28s
##
## Coefficients:
##          Estimate Std. Error z-value Pr(>|z|)
## pf        -0.8733     0.1066   -8.19  2.2e-16 ***
## cl        -0.1718     0.0422   -4.07  4.7e-05 ***
## loc        1.8081     0.2289    7.90  2.9e-15 ***
## wk         1.7543     0.2222    7.89  2.9e-15 ***
## tod       -8.5960     0.9865   -8.71  < 2e-16 ***
## seas      -8.8653     1.0128   -8.75  < 2e-16 ***
## asc2       0.3044     0.1539    1.98   0.0479 *
## asc3       0.1563     0.1598    0.98   0.3279
## asc4       0.1133     0.1568    0.72   0.4698
## sd.cl      0.3643     0.0442    8.25  2.2e-16 ***
## sd.loc     1.1014     0.2738    4.02  5.7e-05 ***
## sd.wk      1.2053     0.2493    4.84  1.3e-06 ***
## sd.tod     1.4655     0.2335    6.28  3.5e-10 ***
## sd.seas    1.8110     0.2958    6.12  9.2e-10 ***
## sd.asc2    0.5264     0.1791    2.94   0.0033 **
## sd.asc3    0.0849     0.2246    0.38   0.7054
## sd.asc4    0.2407     0.1881    1.28   0.2007
## tau        0.6777     0.1490    4.55  5.4e-06 ***
## gamma      0.3625     0.1747    2.08   0.0380 *
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
## Log Likelihood: -729
## Number of observations: 750
## Number of iterations: 98
## Exit of MLE: successful convergence
## Simulation based on 50 draws
```

Since we are including the ASCs as additional variables, the second part of the formula does not include the ASCs (`| 0`). Note also that even though the ASCs are random, they are not scaled: `notscale = c(rep(0, 6), 1, 1, 1)` indicates that the last three variables in the first part of the `formula` (`asc2`, `asc3`, and `asc4`) are not scaled.

Another important issue is that `gmnl` estimates γ directly by default as suggested by Keane and Wasi (2013). However, one can estimate γ^* where $\gamma = \exp(\gamma^*)/(1+\exp(\gamma^*))$ as suggested by Fiebig *et al.* (2010), by specifying `hgamma = "indirect"`. (Thus, `hgamma = "direct"` is the default setting.)

The G-MNL estimation code is also very convenient when one wants to estimate S-MNL models with random effects (Keane and Wasi 2013). In this case, the user can fix γ and use `model = "gmnl"`.

```
Elec.smnl.re <- gmnl(choice ~ pf + cl + loc + wk + tod + seas +
  asc2 + asc3 + asc4 | 0,
  data = Electr,
  subset = 1:3000,
  model = 'gmnl',
  R = 50,
  panel = TRUE,
  print.init =  TRUE,
  notscale = c(rep(0, 6), 1, 1, 1),
  ranp = c(asc2 = "n", asc3 = "n", asc4 = "n"),
  init.gamma = 0,
  fixed = c(rep(FALSE, 16), TRUE),
  correlation =  TRUE)

##
## The following variables are not scaled:
## [1] "asc2" "asc3" "asc4"
##
## Starting Values:
##          pf          cl         loc          wk          tod
##     -0.6018     -0.1350      1.2223      1.0387      -5.3686
##        seas        asc2        asc3        asc4 sd.asc2.asc2
##     -5.5623      0.2097      0.0811      0.1065       0.1000
## sd.asc2.asc3 sd.asc2.asc4 sd.asc3.asc3 sd.asc3.asc4 sd.asc4.asc4
##      0.1000      0.1000      0.1000      0.1000       0.1000
```

```
##          tau        gamma
##       0.1000       0.0000
## Estimating GMNL model

summary(Elec.smnl.re)

##
## Model estimated on: Thu Jun 04 13:16:39 2015
##
## Call:
## gmnl(formula = choice ~ pf + cl + loc + wk + tod + seas + asc2 +
##     asc3 + asc4 | 0, data = Electr, subset = 1:3000, model = "gmnl",
##     ranp = c(asc2 = "n", asc3 = "n", asc4 = "n"), R = 50, correlation = TRUE,
##     panel = TRUE, init.gamma = 0, notscale = c(rep(0, 6), 1,
##         1, 1), print.init = TRUE, fixed = c(rep(FALSE, 16), TRUE),
##     method = "bfgs")
##
## Frequencies of categories:
##
##     1     2     3     4
## 0.215 0.303 0.217 0.265
##
## The estimation took: 0h:0m:16s
##
## Coefficients:
##              Estimate Std. Error z-value Pr(>|z|)
## pf            -0.6299     0.1148   -5.49  4.1e-08 ***
## cl            -0.1377     0.0328   -4.20  2.7e-05 ***
## loc            1.2749     0.2200    5.80  6.8e-09 ***
## wk             1.1119     0.1929    5.76  8.2e-09 ***
## tod           -6.2008     1.0450   -5.93  3.0e-09 ***
## seas          -6.3681     1.0802   -5.90  3.7e-09 ***
## asc2           0.2124     0.1442    1.47   0.1409
## asc3           0.2295     0.1351    1.70   0.0894 .
## asc4           0.1536     0.1310    1.17   0.2410
## sd.asc2.asc2   0.5694     0.2184    2.61   0.0091 **
## sd.asc2.asc3   0.3066     0.1813    1.69   0.0908 .
## sd.asc2.asc4   0.1508     0.1995    0.76   0.4497
## sd.asc3.asc3   0.0445     0.2192    0.20   0.8390
## sd.asc3.asc4   0.0893     0.2188    0.41   0.6832
## sd.asc4.asc4  -0.0406     0.2034   -0.20   0.8419
## tau            1.1009     0.1852    5.94  2.8e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
```

```
## Log Likelihood: -841
## Number of observations: 750
## Number of iterations: 76
## Exit of MLE: successful convergence
## Simulation based on 50 draws
```

The argument `init.gamma` indicates the initial value for γ. In this case we set it at zero. The next step is to set the parameters that are fixed by using the argument `fixed`, which is passed to the `maxLik` function. Note that the user needs to be careful with the order of the parameters. We encourage the user to estimate first a model where all the parameters are freely estimated with the argument `print.init = TRUE`. This argument will display the initial values and the order used by `gmnl`. Generally, γ is the last parameter that enters the likelihood specification. So, by typing `fixed = c(rep(FALSE, 16), TRUE)` we are only holding γ fixed at zero, and the rest of the coefficients are freely estimated.

By default, the initial values for the mean of the random parameters come from an MNL, and the standard deviations or spread are set at 0.1. However, the starting values from an MNL model may not be the best guess, since the G-MNL model is not globally concave. The best starting values for a G-MNL model with correlated parameters might be: 1) G-MNL with uncorrelated parameters, 2) MIXL with correlated parameters, or 3) GMNL with correlated parameters with γ fixed at 0. One can first get these initial parameters and then use the `start` argument of `gmnl` to indicate the vector of appropriate starting values (see Section 3.8 for an example of how to use the `start` argument).

3.7. Estimating LC and MM-MNL models

The next example shows how an LC model with two classes can be estimated:

```
Elec.lc <- gmnl(choice ~ pf + cl + loc + wk + tod + seas| 0 |
  0 | 0 | 1,
  data = Electr,
  subset = 1:3000,
  model = 'lc',
  panel = TRUE,
  Q = 2)

## Estimating LC model
```

Note that for the LC model, one needs to specify at least a constant in the fifth part of the `formula`. If the class assignment w_{iq} is also determined by socio-economic characteristics, those covariates can also be included in the fifth part. The LC model is estimated by typing `model = "lc"`, and the prespecified number of classes is indicated with the argument `Q`.

```
summary(Elec.lc)

##
## Model estimated on: Thu Jun 04 13:16:39 2015
```

```
##
## Call:
## gmnl(formula = choice ~ pf + cl + loc + wk + tod + seas | 0 |
##     0 | 0 | 1, data = Electr, subset = 1:3000, model = "lc",
##     Q = 2, panel = TRUE, method = "bfgs")
##
## Frequencies of categories:
##
##     1     2     3     4
## 0.215 0.303 0.217 0.265
##
## The estimation took: 0h:0m:1s
##
## Coefficients:
##              Estimate Std. Error z-value Pr(>|z|)
## class.1.pf    -0.4458     0.0876   -5.09  3.6e-07 ***
## class.1.cl    -0.1847     0.0301   -6.14  8.3e-10 ***
## class.1.loc    1.2144     0.1618    7.50  6.2e-14 ***
## class.1.wk     0.9641     0.1429    6.75  1.5e-11 ***
## class.1.tod   -3.2184     0.6880   -4.68  2.9e-06 ***
## class.1.seas  -3.4865     0.6929   -5.03  4.9e-07 ***
## class.2.pf    -0.8431     0.0968   -8.71  < 2e-16 ***
## class.2.cl    -0.1242     0.0453   -2.74   0.0061 **
## class.2.loc    1.6445     0.2689    6.12  9.6e-10 ***
## class.2.wk     1.4139     0.2120    6.67  2.6e-11 ***
## class.2.tod   -9.3732     0.8676  -10.80  < 2e-16 ***
## class.2.seas  -9.2647     0.8847  -10.47  < 2e-16 ***
## (class)2      -0.2200     0.0788   -2.79   0.0052 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
## Log Likelihood: -793
## Number of observations: 750
## Number of iterations: 77
## Exit of MLE: successful convergence
```

Finally, the next example estimates an MM-MNL with two mixtures of normal:

```
Elec.mm <- gmnl(choice ~ pf + cl + loc + wk + tod + seas| 0 | 0 | 0 | 1,
  data = Electr,
  subset = 1:3000,
  model = 'mm',
  R = 50,
  panel = TRUE,
  ranp = c(pf = "n", cl = "n", loc = "n",
```

```
          wk = "n", tod = "n", seas = "n"),
  Q = 2,
  iterlim = 500)

## Estimating MM-MNL model

summary(Elec.mm)

##
## Model estimated on: Thu Jun 04 13:17:22 2015
##
## Call:
## gmnl(formula = choice ~ pf + cl + loc + wk + tod + seas | 0 |
##     0 | 0 | 1, data = Electr, subset = 1:3000, model = "mm",
##     ranp = c(pf = "n", cl = "n", loc = "n", wk = "n", tod = "n",
##         seas = "n"), R = 50, Q = 2, panel = TRUE, iterlim = 500,
##     method = "bfgs")
##
## Frequencies of categories:
##
##     1     2     3     4
## 0.215 0.303 0.217 0.265
##
## The estimation took: 0h:0m:42s
##
## Coefficients:
##                    Estimate Std. Error z-value Pr(>|z|)
## class.1.pf         -1.28047    0.12281  -10.43  < 2e-16 ***
## class.1.cl         -0.49715    0.08070   -6.16  7.2e-10 ***
## class.1.loc         0.64457    0.24097    2.67  0.00747 **
## class.1.wk          0.71243    0.21824    3.26  0.00110 **
## class.1.tod       -11.62539    1.02751  -11.31  < 2e-16 ***
## class.1.seas      -12.65728    1.13964  -11.11  < 2e-16 ***
## class.2.pf         -0.43574    0.11588   -3.76  0.00017 ***
## class.2.cl          0.08464    0.08288    1.02  0.30714
## class.2.loc         3.38622    0.38187    8.87  < 2e-16 ***
## class.2.wk          2.72092    0.32689    8.32  < 2e-16 ***
## class.2.tod        -4.71641    1.05415   -4.47  7.7e-06 ***
## class.2.seas       -4.64754    0.96000   -4.84  1.3e-06 ***
## class.1.sd.pf       0.10535    0.03941    2.67  0.00750 **
## class.1.sd.cl       0.26899    0.05833    4.61  4.0e-06 ***
## class.1.sd.loc      0.00759    0.26552    0.03  0.97718
## class.1.sd.wk       0.11417    0.58066    0.20  0.84413
## class.1.sd.tod      2.20521    0.45035    4.90  9.8e-07 ***
## class.1.sd.seas     2.32199    0.45198    5.14  2.8e-07 ***
## class.2.sd.pf       0.19696    0.03372    5.84  5.2e-09 ***
```

```
## class.2.sd.cl     0.35530    0.07492    4.74  2.1e-06 ***
## class.2.sd.loc    0.58709    0.28699    2.05  0.04079 *
## class.2.sd.wk     1.10037    0.30323    3.63  0.00028 ***
## class.2.sd.tod    1.40252    0.51868    2.70  0.00685 **
## class.2.sd.seas   0.07311    0.25655    0.28  0.77565
## (class)2         -0.13719    0.07844   -1.75  0.08028 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
## Log Likelihood: -672
## Number of observations: 750
## Number of iterations: 125
## Exit of MLE: successful convergence
## Simulation based on 50 draws
```

The specification is similar to that of the LC model, but we now allow the parameters in each class to be normally distributed using the argument `ranp`. It is worth mentioning that the number of iterations required for this model is greater than that for previous models. For that reason we have set the maximum of iterations at 500 using the argument `iterlim`.

3.8. Willingness-to-pay space

Willingness-to-pay space models reparameterize the parameter space in such a way that the marginal WTP for each attribute is directly estimated rather than the marginal utility (preference parameters). Train and Weeks (2005) and Sonnier, Ainslie, and Otter (2007) extend the WTP-space approach by allowing random parameters (and, consequently, random willingness-to-pay measures). The WTP-space approach is very appealing because it allows the analyst to estimate the WTP heterogeneity distribution directly (Scarpa, Thiene, and Train 2008).

To illustrate the concept of WTP space, and how it can be estimated using **gmnl**, we will first show the case without random parameters. The standard procedure to derive willingness-to-pay measures is to start with a model in preference space. For example, consider the simple conditional logit model,

```
clogit <- gmnl(choice ~ pf + cl + loc + wk + tod + seas| 0,
  data = Electr,
  subset = 1:3000)
summary(clogit)

##
## Model estimated on: Thu Jun 04 13:17:22 2015
##
## Call:
## gmnl(formula = choice ~ pf + cl + loc + wk + tod + seas | 0,
##     data = Electr, subset = 1:3000, method = "nr")
```

```
##
## Frequencies of categories:
##
##     1     2     3     4
## 0.215 0.303 0.217 0.265
##
## The estimation took: 0h:0m:0s
##
## Coefficients:
##      Estimate Std. Error z-value Pr(>|z|)
## pf    -0.6113     0.0548  -11.15  < 2e-16 ***
## cl    -0.1398     0.0204   -6.85  7.2e-12 ***
## loc    1.1986     0.1197   10.01  < 2e-16 ***
## wk     1.0304     0.1063    9.69  < 2e-16 ***
## tod   -5.4540     0.4341  -12.56  < 2e-16 ***
## seas  -5.6648     0.4419  -12.82  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by Newton-Raphson maximisation
## Log Likelihood: -870
## Number of observations: 750
## Number of iterations: 4
## Exit of MLE: gradient close to zero
```

To estimate the willingness to pay for each attribute, one needs to divide each attribute parameter by that of price `pf`. This ratio can be easily retrieved using the function `wtp.gmnl`:

```
wtp.gmnl(clogit, wrt = "pf")

##
## Willigness-to-pay respect to:  pf
##
##      Estimate Std. Error t-value Pr(>|t|)
## cl     0.2287     0.0358    6.38  1.8e-10 ***
## loc   -1.9610     0.2304   -8.51  < 2e-16 ***
## wk    -1.6858     0.1949   -8.65  < 2e-16 ***
## tod    8.9226     0.2025   44.07  < 2e-16 ***
## seas   9.2675     0.2164   42.83  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The argument `wrt = "pf"` indicates that all the parameters should be divided by the parameter of the attribute `pf`.

Another way to estimate the same WTP coefficients is to use the S-MNL model. We need first to compute the negative of the price coefficient using the `mlogit.data` function:

```
Electr0 <- mlogit.data(Electricity, id = "id", choice = "choice",
  varying = 3:26, shape = "wide", sep = "",
  opposite = c("pf"))
```

Next, we need to set the values for the price parameter and τ at 1 and 0, respectively. The `fixed` argument is used to set these values.

```
start <- c(1, 0, 0, 0, 0, 0, 0, 0)
wtps <- gmnl(choice ~ pf + cl + loc + wk + tod + seas|0 | 0 | 0 | 1,
  data = Electr0,
  model = "smnl",
  subset = 1:3000,
  R = 1,
  fixed = c(TRUE, FALSE, FALSE, FALSE, FALSE,
            FALSE, TRUE, FALSE),
  panel = TRUE,
  start = start,
  method = "bhhh",
  iterlim = 500)

## Estimating SMNL model
```

Note also that we fitted the S-MNL model with a constant in the scale. This constant, after a proper transformation, will represent the price parameter. Since we are working with a fixed parameter model, the number of draws is set equal to 1.

```
summary(wtps)

##
## Model estimated on: Thu Jun 04 13:17:23 2015
##
## Call:
## gmnl(formula = choice ~ pf + cl + loc + wk + tod + seas | 0 |
##     0 | 0 | 1, data = Electr0, subset = 1:3000, model = "smnl",
##     start = start, R = 1, panel = TRUE, fixed = c(TRUE, FALSE,
##         FALSE, FALSE, FALSE, FALSE, TRUE, FALSE), method = "bhhh",
##     iterlim = 500)
##
## Frequencies of categories:
##
##     1     2     3     4
## 0.215 0.303 0.217 0.265
##
## The estimation took: 0h:0m:1s
##
## Coefficients:
```

```
##                   Estimate Std. Error z-value Pr(>|z|)
## cl                 -0.2287     0.0361   -6.34  2.4e-10 ***
## loc                 1.9610     0.2284    8.59  < 2e-16 ***
## wk                  1.6858     0.1915    8.80  < 2e-16 ***
## tod                -8.9226     0.2025  -44.06  < 2e-16 ***
## seas               -9.2675     0.2166  -42.79  < 2e-16 ***
## het.(Intercept)  -0.4922     0.0917   -5.37  7.9e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BHHH maximisation
## Log Likelihood: -870
## Number of observations: 750
## Number of iterations: 16
## Exit of MLE: successive function values within tolerance limit
## Simulation based on 1 draws
```

Each value in the output represents the WTP estimates for each respective attribute. Note that these WTP estimates are the same as those obtained using the `wtp.gmnl` function. The price coefficient can be obtained using the following transformation:

```
- exp(coef(wtps)["het.(Intercept)"])

## het.(Intercept)
##          -0.611
```

If one requires the standard error for the price coefficient the `deltamethod` function from the **msm** (Jackson 2011) package can be used in the following way:

```
library("msm")
estmean <- coef(wtps)
estvar  <- vcov(wtps)
se <- deltamethod(~ - exp(x6), estmean, estvar, ses = TRUE)
se

## [1] 0.056
```

Using the same idea, one can let the WTP to vary across individuals. To do so, we can estimate a G-MNL where the parameter of price and γ are fixed as in the previous example:

```
start2 <- c(1, coef(wtps), rep(0.1, 5), 0.1, 0)
wtps2 <- gmnl(choice ~ pf + cl + loc + wk + tod + seas|0 | 0 | 0 | 1,
  data = Electr0,
  subset = 1:3000,
  model = "gmnl",
  R = 50,
```

```
  fixed = c(TRUE, rep(FALSE, 12), TRUE),
  panel = TRUE,
  start = start2,
  ranp = c(cl = "n", loc = "n", wk = "n", tod = "n", seas = "n"))

## Estimating GMNL model

summary(wtps2)

##
## Model estimated on: Thu Jun 04 13:17:54 2015
##
## Call:
## gmnl(formula = choice ~ pf + cl + loc + wk + tod + seas | 0 |
##     0 | 0 | 1, data = Electr0, subset = 1:3000, model = "gmnl",
##     start = start2, ranp = c(cl = "n", loc = "n", wk = "n", tod = "n",
##         seas = "n"), R = 50, panel = TRUE, fixed = c(TRUE, rep(FALSE,
##         12), TRUE), method = "bfgs")
##
## Frequencies of categories:
##
##     1     2     3     4
## 0.215 0.303 0.217 0.265
##
## The estimation took: 0h:0m:31s
##
## Coefficients:
##                   Estimate Std. Error z-value Pr(>|z|)
## cl                 -0.2728     0.0518   -5.26  1.4e-07 ***
## loc                 2.1631     0.2452    8.82  < 2e-16 ***
## wk                  1.9424     0.1935   10.04  < 2e-16 ***
## tod                -9.6782     0.2933  -32.99  < 2e-16 ***
## seas               -9.8865     0.2772  -35.67  < 2e-16 ***
## het.(Intercept)     0.1142     0.1400    0.82     0.41
## sd.cl               0.4115     0.0520    7.91  2.7e-15 ***
## sd.loc              1.7849     0.2515    7.10  1.3e-12 ***
## sd.wk               1.2865     0.2212    5.81  6.1e-09 ***
## sd.tod              1.7174     0.2478    6.93  4.2e-12 ***
## sd.seas             2.2144     0.3722    5.95  2.7e-09 ***
## tau                 0.6904     0.1394    4.95  7.3e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by BFGS maximisation
## Log Likelihood: -737
## Number of observations: 750
```

```
## Number of iterations: 122
## Exit of MLE: successful convergence
## Simulation based on 50 draws
```

3.9. Individual parameters

Similarly to the **Rchoice** package (Sarrias 2015), **gmnl** also allows the analyst to get the conditional estimates for each individual in the sample (see for example Train 2009; Greene 2012). Using Bayes' theorem we obtain

$$f(\boldsymbol{\beta}_i|\mathbf{y}_i,\mathbf{X}_i,\boldsymbol{\theta}) = \frac{f(\mathbf{y}_i|\mathbf{X}_i,\boldsymbol{\beta}_i)g(\boldsymbol{\beta}_i|\boldsymbol{\theta})}{\int_{\boldsymbol{\beta}_i} f(\mathbf{y}_i|\mathbf{X}_i,\boldsymbol{\beta}_i)g(\boldsymbol{\beta}_i|\boldsymbol{\theta})d\boldsymbol{\beta}_i},$$

where $f(\boldsymbol{\beta}_i|\mathbf{y}_i,\mathbf{X}_i,\boldsymbol{\theta})$ is the distribution of the individual parameters $\boldsymbol{\beta}_i$ conditional on the observed sequence of choices, and $g(\boldsymbol{\beta}_i|\boldsymbol{\theta})$ is the unconditional distribution. The conditional expectation of $\boldsymbol{\beta}_i$ is thus given by:

$$\mathsf{E}\left[\boldsymbol{\beta}_i|\mathbf{y}_i,\mathbf{X}_i,\boldsymbol{\theta}\right] = \frac{\int_{\boldsymbol{\beta}_i} \boldsymbol{\beta}_i f(\mathbf{y}_i|\mathbf{X}_i,\boldsymbol{\beta}_i)g(\boldsymbol{\beta}_i|\boldsymbol{\theta})}{\int_{\boldsymbol{\beta}_i} f(\mathbf{y}_i|\mathbf{X}_i,\boldsymbol{\beta}_i)g(\boldsymbol{\beta}_i|\boldsymbol{\theta})d\boldsymbol{\beta}_i}. \tag{5}$$

The expectation in Equation 5 gives us the conditional mean of the distribution of the random parameters, which can also be interpreted as the posterior distribution of the individual parameters. Simulators for this conditional expectation are presented below, for both the continuous and discrete cases:

$$\begin{aligned}
\widehat{\widehat{\boldsymbol{\beta}}}_i = \widehat{\mathsf{E}}\left[\boldsymbol{\beta}_i|\mathbf{y}_i,\mathbf{X}_i,\boldsymbol{\theta}\right] &= \frac{\frac{1}{R}\sum_{r=1}^{R}\widehat{\boldsymbol{\beta}}_{ir}\prod_t f(y_{it}|\mathbf{x}_{it},\widehat{\boldsymbol{\beta}}_{ir},\widehat{\boldsymbol{\theta}})}{\frac{1}{R}\sum_{r=1}^{R}\prod_t f(y_{it}|\mathbf{x}_{it},\widehat{\boldsymbol{\beta}}_{ir},\widehat{\boldsymbol{\theta}})} \\
\widehat{\widehat{\boldsymbol{\beta}}}_i = \widehat{\mathsf{E}}\left[\boldsymbol{\beta}_i|\mathbf{y}_i,\mathbf{X}_i,\boldsymbol{\theta}_q\right] &= \frac{\sum_{q=1}^{Q}\widehat{\boldsymbol{\beta}}_q\widehat{w}_{iq}\prod_t f(y_{it}|\mathbf{x}_{it},\widehat{\boldsymbol{\beta}}_{ir},\widehat{\boldsymbol{\theta}}_q)}{\sum_{q=1}^{Q}\prod_t f(y_{it}|\mathbf{x}_{it},\widehat{\boldsymbol{\beta}}_q,\widehat{\boldsymbol{\theta}}_q)}
\end{aligned}$$

In order to construct the confidence interval for $\widehat{\widehat{\boldsymbol{\beta}}}_i$, we can derive an estimator of the conditional variance from the point estimates as follows (Greene 2012, chap. 15):

$$\widehat{V}_i = \widehat{\mathsf{E}}\left[\boldsymbol{\beta}_i^2|\mathbf{y}_i,\mathbf{X}_i,\boldsymbol{\theta}\right] - \widehat{\mathsf{E}}\left[\boldsymbol{\beta}_i|\mathbf{y}_i,\mathbf{X}_i,\boldsymbol{\theta}\right]^2.$$

An approximate normal-based 95% confidence interval can be then constructed as $\widehat{\widehat{\boldsymbol{\beta}}}_i \pm 1.96 \times \widehat{V}_i^{1/2}$.

The **gmnl** package uses these formulae to compute the individual parameters along with their 95% confidence interval. As an illustration, we can plot the kernel density of the individuals' conditional mean for the `loc` parameter using `Elec.cor` model by typing the following:

```
plot(Elec.cor, par = "loc", effect = "ce", type = "density",
  col = "grey")
```

Figure 1 displays the distribution of the individuals' conditional mean for the parameter of `loc`. The gray area gives us the proportion of individuals with a positive conditional mean.

The 95% confidence interval of the conditional mean for the first 30 individuals is shown in Figure 2, which was plotted using the following syntax:[5]

```
plot(Elec.cor, par = "loc", effect = "ce", ind = TRUE, id = 1:30)
```

Another important function in **gmnl** is `effect.gmnl`. This function allows the users to get the individuals' conditional mean of both the preference parameters and the willingness-to-pay measures. For example, one can plot the individual conditional mean and standard errors (Figure 2) by typing:

```
bi.loc <- effect.gmnl(Elec.cor, par = "loc", effect = "ce")
summary(bi.loc$mean)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   -0.79    0.42    2.04    2.12    3.46    7.13

summary(bi.loc$sd.est)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.113   0.564   0.795   0.866   1.130   1.860
```

The conditional mean of the willingness to pay for "`loc`" (wtp $= \beta_{i,\texttt{loc}}/\beta_{pf}$) for all individuals in the sample can be obtained using:

```
wtp.loc <- effect.gmnl(Elec.cor, par = "loc", effect = "wtp", wrt = "pf")
```

Note that the argument `par` is the variable whose parameter goes in the numerator, and the argument `wrt` is a string indicating which parameter goes in the denominator.

```
summary(wtp.loc$mean)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   -8.19   -3.98   -2.35   -2.44   -0.48    0.91

summary(wtp.loc$sd.est)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.130   0.648   0.914   0.996   1.300   2.130
```

4. Conclusions

[5] **gmnl** uses **plotrix** package (Lemon 2006) to create the confidence interval graph.

The package **gmnl** implements the maximum likelihood estimator of random parameter logit models with heterogeneity distributions that can be continuous, discrete, or discrete-continuous mixtures. In this paper we have shown how **gmnl** can fit several extensions to the standard multinomial logit model, including the recently derived mixed-mixed multinomial logit (MM-MNL). To our knowledge there is no other widely available statistical package that has implemented the maximum simulated likelihood estimator of MM-MNL, and we want to highlight that **gmnl** makes use of analytical expressions of the gradient. **gmnl** is also the first implementation in R of the estimator of the scale heterogeneity multinomial logit (S-MNL), the generalized multinomial logit (G-MNL), and the latent class logit (LC). Whereas there are other packages in R for the estimation of MIXL, **gmnl** allows for the inclusion of individual-specific variables to explain the mean of the random parameters for a mixture of deterministic taste variations and unobserved preference heterogeneity. In addition, **gmnl** also implements Johnson S_b heterogeneity distributions.

Another key post-estimation functionality of **gmnl** that we have illustrated in this paper is the derivation of conditional point and interval estimates of either the random parameters or willingness-to-pay measures at the individual level. Random parameter models can be used to make inference on the preference parameters of each individual in the sample, but most packages that estimate MIXL models lack a command to produce individual-level estimates. **gmnl** is able to compute individual parameters for all generalized logit models that are implemented in the package, including G-MNL, MIXL, and LC.

Additional functionalities that we expect to incorporate in the future are the consideration of different choice sets for each individual and the implementation of different methods for the construction of confidence intervals of willingness-to-pay measures.

References

Boxall PC, Adamowicz WL (2002). "Understanding Heterogeneous Preferences in Random Utility Models: A Latent Class Approach." *Environmental and Resource Economics*, **23**(4), 421–446.

Bujosa A, Riera A, Hicks RL (2010). "Combining Discrete and Continuous Representations of Preference Heterogeneity: A Latent Class Approach." *Environmental and Resource Economics*, **47**(4), 477–493.

Croissant Y (2012). "Estimation of Multinomial Logit Models in R: The **mlogit** Packages." *R package version 0.2-2*. URL http://cran.r-project.org/web/packages/mlogit/vignettes/mlogit.pdf.

Dumont J, Keller J, Carpenter C (2014). ***RSGHB**: Functions for Hierarchical Bayesian Estimation: A Flexible Approach.* R package version 1.0.2, URL http://CRAN.R-project.org/package=RSGHB.

Fiebig DG, Keane MP, Louviere J, Wasi N (2010). "The Generalized Multinomial Logit Model: Accounting for Scale and Coefficient Heterogeneity." *Marketing Science*, **29**(3), 393–421.

Gourieroux C, Monfort A (1997). *Simulation-based Econometric Methods*. Oxford University Press.

Greene WH (2012). *Econometric Analysis*. 7th edition. Prentice Hall.

Greene WH, Hensher DA (2003). "A Latent Class Model for Discrete Choice Analysis: Contrasts With Mixed Logit." *Transportation Research Part B: Methodological*, **37**(8), 681–698.

Greene WH, Hensher DA (2010). "Does Scale Heterogeneity Across Individuals Matter? An Empirical Assessment of Alternative Logit Models." *Transportation*, **37**(3), 413–428.

Greene WH, Hensher DA (2013). "Revealing Additional Dimensions of Preference Heterogeneity in a Latent Class Mixed Multinomial Logit Model." *Applied Economics*, **45**(14), 1897–1902.

Hajivassiliou VA, Ruud PA (1986). "Classical Estimation Methods for LDV Models Using Simulation." In RF Engle, D McFadden (eds.), *Handbook of Econometrics*, volume 4 of *Handbook of Econometrics*, chapter 40, pp. 2383–2441. Elsevier.

Hasan A, Zhiyu W, Mahani AS (2015). ***mnlogit**: Multinomial Logit Model.* R package version 1.2.1, URL http://CRAN.R-project.org/package=mnlogit.

Henningsen A, Toomet O (2011). "**maxLik**: A Package for Maximum Likelihood Estimation in R." *Computational Statistics*, **26**(3), 443–458.

Hensher DA, Greene WH (2003). "The Mixed Logit Model: The State of Practice." *Transportation*, **30**(2), 133–176.

Hess S, Ben-Akiva M, Gopinath D, Walker J (2011). "Advantages of Latent Class Over Continuous Mixture of Logit Models." *Working paper*, Institute for Transport Studies, University of Leeds.

Hess S, Rose JM (2012). "Can Scale and Coefficient Heterogeneity be Separated in Random Coefficients Models?" *Transportation*, **39**(6), 1225–1239.

Hess S, Stathopoulos A (2013). "Linking Response Quality to Survey Engagement: A Combined Random Scale and Latent Variable Approach." *Journal of Choice Modelling*, **7**, 1–12.

Imai K, Dyk DAV (2005). "**MNP**: R Package for Fitting the Multinomial Probit Model." *Journal of Statistical Software*, **14**(3), 1–32. URL http://www.jstatsoft.org/v14/i03.

Jackson CH (2011). "Multi-State Models for Panel Data: The **msm** Package for R." *Journal of Statistical Software*, **38**(8), 1–29. URL http://www.jstatsoft.org/v38/i08/.

Keane M, Wasi N (2013). "Comparing Alternative Models of Heterogeneity in Consumer Choice Behavior." *Journal of Applied Econometrics*, **28**(6), 1018–1045.

Kleiber C, Zeileis A (2008). *Applied Econometrics with R*. Springer-Verlag, New York. URL http://CRAN.R-project.org/package=AER.

Lee LF (1992). "On Efficiency of Methods of Simulated Moments and Maximum Simulated Likelihood Estimation of Discrete Response Models." *Econometric Theory*, **8**(04), 518–552.

Leisch F (2004). "**FlexMix**: A General Framework for Finite Mixture Models and Latent Class Regression in R." *Journal of Statistical Software*, **11**(8), 1–18. URL http://www.jstatsoft.org/v11/i08/.

Lemon J (2006). "**Plotrix**: A Package in the Red Light District of R." *R-News*, **6**(4), 8–12.

Linzer DA, Lewis JB (2011). "**poLCA**: An R Package for Polytomous Variable Latent Class Analysis." *Journal of Statistical Software*, **42**(10), 1–29. URL http://www.jstatsoft.org/v42/i10/.

McFadden D (1974). "Conditional Logit Analysis of Qualitative Choice Behavior." In P Zarembka (ed.), *Frontiers in Econometrics*, pp. 105–142. Academic Press, New York.

McFadden D, Train K (2000). "Mixed MNL Models for Discrete Response." *Journal of Applied Econometrics*, **15**(5), 447–470.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org.

Rossi P (2012). ***bayesm**: Bayesian Inference for Marketing/Micro-econometrics*. R package version 2.2-5, URL http://CRAN.R-project.org/package=bayesm.

Sarrias M (2015). ***Rchoice**: Discrete Choice (Binary, Poisson and Ordered) Models with Random Parameters*. R package version 0.3, URL http://CRAN.R-project.org/package=Rchoice.

Sarrias M, Daziano R (2015). ***gmnl**: Multinomial Logit Models with Random Parameters*. R package version 1.1, URL http://CRAN.R-project.org/package=gmnl.

Scarpa R, Thiene M (2005). "Destination Choice Models for Rock Climbing in the Northeastern Alps: A Latent-class Approach Based on Intensity of Preferences." *Land Economics*, **81**(3), 426–444.

Scarpa R, Thiene M, Train K (2008). "Utility in Willingness to Pay Space: A Tool to Address Confounding Random Scale Effects in Destination Choice to the Alps." *American Journal of Agricultural Economics*, **90**(4), 994–1010.

Shen J (2009). "Latent Class Model or Mixed Logit Model? A Comparison by Transport Mode Choice Data." *Applied Economics*, **41**(22), 2915–2924.

Sonnier G, Ainslie A, Otter T (2007). "Heterogeneity Distributions of Willingness-to-Pay in Choice Models." *Quantitative Marketing and Economics*, **5**(3), 313–331.

Train K (2009). *Discrete Choice Methods with Simulation*. 2nd edition. Cambridge University Press.

Train K, Weeks M (2005). "Discrete Choice Models in Preference Space and Willingness-to-Pay Space." In R Scarpa, A Alberini (eds.), *Applications of Simulation Methods in Environmental and Resource Economics*, volume 6 of *The Economics of Non-Market Goods and Resources*, pp. 1–16. Springer Netherlands. ISBN 978-1-4020-3683-5. doi:10.1007/1-4020-3684-1_1. URL http://dx.doi.org/10.1007/1-4020-3684-1_1.

Train KE (2008). "EM Algorithms for Nonparametric Estimation of Mixing Distributions." *Journal of Choice Modelling*, **1**(1), 40–69.

Trautmann H, Steuer D, Mersmann O, Bornkamp B (2014). ***truncnorm**: Truncated Normal Distribution*. R package version 1.0-7, URL http://CRAN.R-project.org/package=truncnorm.

Venables WN, Ripley BD (2002). *Modern Applied Statistics with S*. Fourth edition. Springer-Verlag, New York. ISBN 0-387-95457-0, URL http://www.stats.ox.ac.uk/pub/MASS4.

Yee TW (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. URL http://www.jstatsoft.org/v34/i01/.

Zeileis A, Hothorn T (2002). "Diagnostic Checking in Regression Relationships." *R News*, **2**(3), 7–10. URL http://CRAN.R-project.org/doc/Rnews/.

Affiliation:

Mauricio Sarrias
Department of City and Regional Panning
Cornell University
325 W. Sibley Hall, Ithaca, NY 14853, USA

E-mail: msarrias86@gmail.com
URL: https://msarrias.weebly.com

Ricardo A. Daziano
School of Civil and Environmental Engineering
Cornell University
305 Hollister Hall, Ithaca, NY 14850, USA
E-mail : daziano@cornell.edu
URL: http://www.cee.cornell.edu/research/groups/daziano/

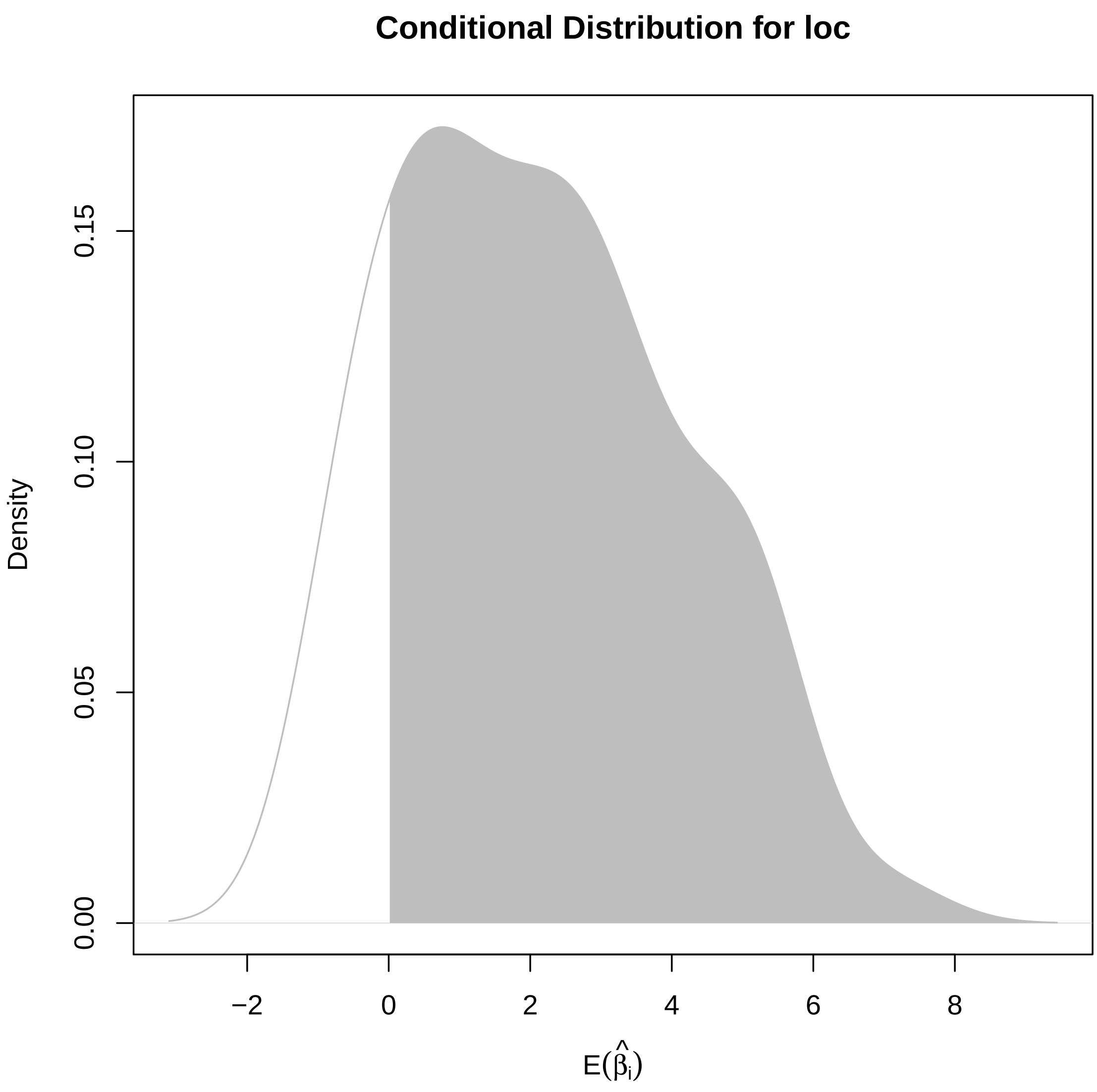

Figure 1: Kernel density of the individuals' conditional mean.

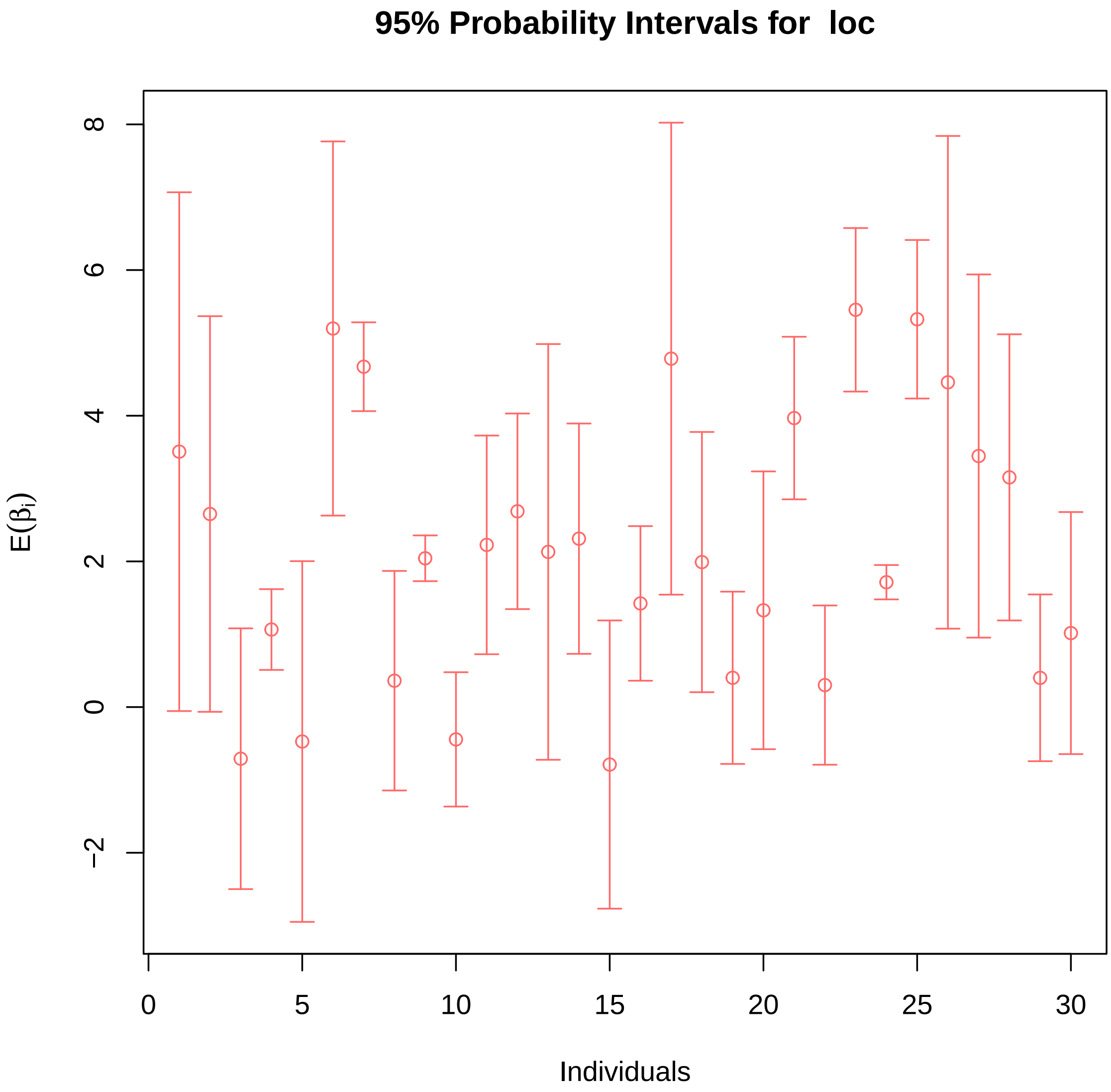

Figure 2: 95% confident interval for the conditional means.