Estimation of discrete choice models with Biogeme

Michel Bierlaire*

November 2, 2015

Report TRANSP-OR xxxxxx
Transport and Mobility Laboratory
School of Architecture, Civil and Environmental Engineering
Ecole Polytechnique Fédérale de Lausanne
transp-or.epfl.ch

*Ecole Polytechnique Fédérale de Lausanne, Transport and Mobility Laboratory, CH-1015 Lausanne, Switzerland. Email: michel.bierlaire@epfl.ch

Contents

Chapter 1

Introduction

Bierlaire Optimization toolbox for GEv Model Estimation (BIOGEME) is a freeware package designed for the development of research in the context of discrete choice models. Originally developed specifically for Multivariate Extreme Value models (called Generalized Extreme Valur, or GEV models, by McFadden, 1978), it can now handle a wide variety of models. The distribution of BIOGEME, as well as related material, is maintained at

`biogeme.epfl.ch`.

The archives of the users group can be found at

`groups.yahoo.com/group/biogeme`

The following pieces of software are available:

Biogeme performs maximum likelihood estimation of the parameters of a list of predetermined models: logit, nested logit, cross nested logit, network MEV, binary probit, ordered logit, and continuous and discrete mixtures of these models. The model description is performed using a simple syntax in a text file. The syntax has been developed using the Bison parser generator. Therefore, we refer sometimes to it as *BisonBiogeme*, as opposed to *PythonBiogeme* presented below.

Biosim performs a sample enumeration of an estimated model on a data set. It applies the model to each observation, and compute the utilities as well as the choice probability of each alternative.

Biomerge is a simple facility designed to merge files with the same number of rows, so that each row is merged with the corresponding rows of the other files. It is typically used to merge a sample file with a file generated by biosim.

PythonBiogeme performs maximum likihood estimation of the parameters of any model such that it is possible to write the likelihood function. It comes with a list of models already implemented, including the logit, the nested logit, the cross nested logit, the MEV models, as well as examples about the specification of other models. The model description is based on an extension of the Python programming language. PythonBiogeme can exploit multi-threading to speed-up the estimation time in the presence of multiple processors.

as well as the following utilities

histograms an utility written in Python that takes a list of raw numbers and organizes them into bins in order to plot histograms.

mod2py a script that runs Biogeme to transform a model written for BisonBiogeme (a `.mod` file) into a model written for PythonBiogeme (a `.py` file).

Likelihood ratio test an excel worksheet that performs a likelihood ratio test.

CNL correlation a Matlab code to compute the correlation matrix of a cross-nested logit model given the value of its parameters.

Variance computation Excel sheet to compute the variance of the difference and the ratio of two random variables.

Prepare data Biogeme requires that the data file contains only numerical data. If your data file happens to contain text data (such as names of cities, for example), this script written in Python will assign a value to each of the strings, generate a data file compliant with Biogeme's requirements as well as a glossary explaning which number corresponds to which string.

In addition to this document, several examples are posted online. Make sure to visit the webpage. If the document, the examples and and your "trial-and-errors" do not help solving problems in using the software, make sure you visit the archives of the users group. Several questions have been asked by other users, and the responses are often available in the archives. If you don't find a solution to your problem, post a message to

groups.yahoo.com/group/biogeme/post

Also, your contribution is very appreciated. If you have suggestions to improve Biogeme, post them to the users group as well.

Biogeme is developed under the GNU environment (www.gnu.org). Therefore, it is easy to install from sources on a Linux platform, as well as on a Apple Mac OS X platform with Xcode (developer.apple.com/xcode/). Executables for Windows are available too. Note that PythonBiogeme running on Windows is based on Cygwin (www.cygwin.com), which significantly slows down the execution. In principle, BisonBiogeme should not suffer from the same limitation and seems to run fine on Windows. In any case, except if you need to run simple examples for teaching purposes, we strongly recommend to use Biogeme on Linux or Mac OS X.

Also, a stand alone version of BisonBiogeme with a graphical user interface is made available for teaching purposes. It is however advised to run the software from a terminal.

Biogeme is distributed free of charge. We ask each user

1. to register to the Biogeme's users group

   groups.yahoo.com/group/biogeme,

   and

2. to mention explicitly the use of the package when publishing results, using a reference to this document as well as the following:

   - Bierlaire, M. (2003). BIOGEME: A free package for the estimation of discrete choice models , Proceedings of the 3rd Swiss Transportation Research Conference, Ascona, Switzerland.
   - Bierlaire, M., and Fetiarison, M. (2009). Estimation of discrete choice models: extending BIOGEME. Proceedings of the 9th Swiss Transport Research Conference, Ascone, Switzerland.

If you have any question about Biogeme 2.4, post them on the users' group.

1.1 Discrete choice models

This section is not an introduction to discrete choice models. Instead, it describes the models that are implemented in Biogeme 2.4. We refer the reader to the abundant literature on the topic, including Ben-Akiva and Lerman (1985) and Train (2009).

Random utility models are based on the following ingredients:

- a finite population with individuals denoted by n, each of them characterized by a vector of socio-economic characteristics s_n;
- for each individual n, a finite choice set $\mathcal{C}_n$ containing J_n alternatives, each of them characterized by a vector of attributes z_{in};
- for each individual n and each alternative $i \in \mathcal{C}_n$, a utility function represented by the random variable

$$U_{in} = V_{in} + \varepsilon_{in}, \tag{1.1}$$

  where $V_{in} = V_{in}(s_n, z_{in}; \theta)$ is the deterministic part, that depends on variables s_n and z_{in}, and unknown parameters θ to be estimated from data, and ε_{in} is the error term, assumed to follow a given distribution, possibly depending on unknown parameters to be estimated too.

The probability that individual n chooses alternative i within choice set $\mathcal{C}_n$ is given by

$$P(i|\mathcal{C}_n) = \Pr(U_{in} \geq U_{jn} \forall j \in \mathcal{C}_n). \tag{1.2}$$

If F is the CDF of the random variable ε_{in}, the choice probability is

$$P(i|\mathcal{C}_n) = \int_{\xi=-\infty}^{+\infty} \frac{\partial F}{\partial \varepsilon_i}(\ldots, V_{in} - V_{(i-1)n} + \xi, \xi, V_{in} - V_{(i+1)n} + \xi, \ldots) d\xi. \tag{1.3}$$

In the following, we drop the index n for notational simplicity.

Multivariate Extreme Value models

Biogeme has been designed for Multivariate Extreme Value (MEV) models (introduced as Generalized Extreme Value models by McFadden, 1978). Given a choice probability generating function $G : \mathbb{R}_+^J \to \mathbb{R}$ verifying the following properties

1. G is homogeneous of degree $\mu > 0$, that is $G(\alpha y) = \alpha^\mu G(y)$ for each $\alpha \neq 0$,
2. $\lim_{y_i \to +\infty} G(y_1, \ldots, y_i, \ldots, y_J) = +\infty$, for each $i = 1, \ldots, J$,
3. the kth partial derivative with respect to k distinct y_i is non-negative if k is odd and non-positive if k is even that is, for any distinct indices $i_1, \ldots, i_k \in \{1, \ldots, J\}$, we have

$$(-1)^k \frac{\partial^k G}{\partial x_{i_1} \ldots \partial x_{i_k}}(x) \leq 0, \ \forall x \in \mathbb{R}_+^J. \tag{1.4}$$

The choice probability is then given by

$$P(i|C) = \frac{e^{V_i + \ln G_i(e^{V_1}, \ldots, e^{V_J})}}{\sum_{j=1}^J e^{V_j + \ln G_j(e^{V_1}, \ldots, e^{V_J})}}, \tag{1.5}$$

where $G_i = \partial G / \partial y_i$.

Biogeme 2.4 implements four instances of that family of discrete choice models and through the use of random coefficients, to normal or uniform mixture of each version.

1. the logit model. If you are using Biogeme, we safely assume that you are familiar with this model. If not, read Ben-Akiva and Lerman (1985). The G function is

$$G(y) = \sum_{i=1}^J y_i^\mu,$$

where μ is a scale parameter. As it is unidentified, it is usually normalized to 1. The choice probability is

$$P(i|\mathcal{C}) = \frac{e^{\mu V_i}}{\sum_{j \in \mathcal{C}} e^{\mu V_j}}, \tag{1.6}$$

2. the nested logit model (Ben-Akiva, 1973, Daly, 1987), where the choice set is partitioned into M nests, so that each alternative belongs to exactly one nest[1]. The G function is

$$G(y) = \sum_{m=1}^M \left(\sum_{\ell \in \mathcal{C}} (\alpha_{\ell m} y_\ell)^{\mu_m} \right)^{\mu/\mu_m}. \tag{1.7}$$

[1] In Biogeme, a nested logit model contains only one level of nests. However, it is possible to handle multiple levels thanks to the Network MEV model (see below).

where $\alpha_{\ell m}$ is 1 if alternative i belongs to nest m and 0 otherwise, μ is a scale parameter usually normalized to 1, and μ_m are nested specific parameters verifying $\mu \leq \mu_m$ for all m. The choice probability is given by

$$P(i|\mathcal{C}) = \frac{e^{\mu_m V_i}}{\sum_{j\in\mathcal{C}} \alpha_{jm} e^{\mu_m V_{jn}}} \frac{\left(\sum_{\ell\in\mathcal{C}} \alpha_{\ell m} e^{\mu_m V_{\ell n}}\right)^{\frac{\mu}{\mu_m}}}{\sum_{p=1}^{M} \left(\sum_{\ell\in\mathcal{C}} \alpha_{\ell p} e^{\mu_p V_{\ell n}}\right)^{\frac{\mu}{\mu_p}}}.$$

3. The cross-nested logit model generalized the nested logit formulation in allowing the α parameters to take values between 0 and 1 in the nested logit formultion, so that an alternative can belong to more than one nest, so that complex correlation structures can be investigated. We refer the reader to the literature for the description of the cross-nested logit model (Small, 1987, Vovsha, 1997, Ben-Akiva and Bierlaire, 2003, Papola, 2004, Bierlaire, 2006, Wen and Koppelman, 2001, Abbe et al., 2007).

$$G(y_1, \ldots, y_J) = \sum_{m=1}^{M} \left(\sum_{j\in\mathcal{C}} (\alpha_{jm}{}^{1/\mu} y_j)^{\mu_m} \right)^{\frac{\mu}{\mu_m}}, \tag{1.8}$$

with $\mu \leq \mu_m$ for all m, and $\alpha_{jm} \geq 0$ for all j and m. For identification purposes, the constraint

$$\sum_{m=1}^{M} \alpha_{jm} = 1 \quad \forall j \in \mathcal{C}$$

must be imposed (see Wen and Koppelman, 2001 and Abbe et al., 2007).

4. the network MEV model. This family of models is equivalent to the Recursive Nested Extreme Value Model proposed by Daly (2001), but based purely on a network structure, as proposed by Bierlaire (2002). We refer the reader to Daly and Bierlaire (2006) for details about this model (called "network GEV" in the paper, following the original naming of McFadden, 1978). It allows to define a wide class of MEV models by designing a network structure, with specific properties, easy to verify. The Logit model, the nested logit model and the cross-nested logit model are special instances of the network MEV model. In the network, there is a parameter associated with each node and with each arc.

   If i is a node corresponding to an alternative,

$$G^i(y_i) = y_i^{\mu_i} \quad i = 1, \ldots, J,$$

and if not,

$$G^i(y) = \sum_{j \in \text{succ}(i)} \alpha_{ij} G^j(y)^{\frac{\mu_i}{\mu_j}},$$

where $\text{succ}(i)$ denotes the set of successors of i, $\mu_i > 0$, $\mu_j > 0$, $\mu_i \leq \mu_j$, $\alpha_{ij} \geq 0$,

Mixtures

Biogeme 2.4 allows to include random parameters in the specification of a MEV model. Suppose that the choice model is based on a vector of fixed parameters θ, and a vector of random parameters η with pdf $f_\eta(\xi;\gamma)$, where γ are the parameter of the distribution, usually referred to as *deep* parameters. The choice probability is then given by

$$P(i|\mathcal{C};\theta,\gamma) = \int_\xi P(i|\mathcal{C};\theta,\xi) f_\eta(\xi;\gamma) d\xi, \tag{1.9}$$

where $P(i|\mathcal{C};\theta,\xi)$ is a MEV model. The model (1.9) is called a *mixture* of MEV models. The integral has no closed form and must be estimated by Monte-Carlo simulation.

Utility function

In BisonBiogeme, the population is assumed to be divided into groups, characterized by the membership function $s(n)$. The specification of the utility function is

$$U_n = \lambda_{s(n)} f(X_n;\beta_f,\beta_N,\beta_U) + \nu_n, \tag{1.10}$$

where

- $X_n \in \mathbb{R}^{J_n \times L}$ is a matrix such that each row $j = 1,\ldots,J_n$ contains both the attributes z_{jn} of each alternative j perceived by individual n, and the socio-economic characteristics s_n of individual n,
- β_f is a vector of fixed, real, parameters,
- $\beta_N \sim N(\beta_0, \Gamma\Gamma^T)$ is a random vector of dimension K, normally distributed with mean $\beta_0 \in \mathbb{R}^K$ and variance-covariance matrix $\Sigma_\beta = \Gamma\Gamma^T \in \mathbb{R}^{K\times K}$,

- β_U is a vector of independent uniformly distributed parameters, such that $(\beta_U)_i \sim U(\beta_{ai} - \beta_{bi}, \beta_{ai} + \beta_{bi})$.
- $f : \mathbb{R}^{J_n \times L} \times \mathbb{R}^K \to \mathbb{R}^J_n$ is a continuously differentiable nonlinear function,
- $\lambda_{s(n)}$ is a scale factor associated with the market segment $s(n)$,
- ν_n is a generalized extreme value distributed random vector with joint cumulative distribution function

$$F(\nu_n) = e^{-G_\gamma(e^{-(\nu_n)_1},\ldots,e^{-(\nu_n)_J})} \tag{1.11}$$

  induced by a generating function $G_\gamma : \mathbb{R}^J_+ \to \mathbb{R}$ verifying conditions 1–3.

We expand the random vector β_N as follows:

$$\beta_N = \beta_0 + \Gamma\xi, \tag{1.12}$$

where $\xi \sim N(0, I)$. Similarly, each uniformly distributed parameter is expanded as follows:

$$(\beta_U)_i = (\beta_a)_i + (\beta_b)_i * \omega_i \tag{1.13}$$

where $\omega_i \sim U(-1, 1)$.

We rewrite (1.10) as:

$$U_n = \lambda_{s(n)} f(X_n; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \xi, \omega) + \nu_n. \tag{1.14}$$

In the following, we will denote

$$V_n = \lambda_{s(n)} f(X_n; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \xi, \omega) \tag{1.15}$$

and

$$V_{in} = \lambda_{s(n)} f(X_n; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \xi, \omega)_i \tag{1.16}$$

where $f(\cdot)_i$ is the ith component of f.

The final form of the probability model can be written by first deriving the probability model conditional on ξ and ω (see McFadden, 1978) using the MEV theory.

$$P(i|\mathcal{C}_n, X_n; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \lambda, \gamma; \xi, \omega) = \frac{e^{V_{in} + \log G_i(\ldots)}}{\sum_{j \in \mathcal{C}_n} e^{V_{jn} + \log G_j(\ldots)}}, \tag{1.17}$$

where $\lambda \in \mathbb{R}^S$ is the vector of scale factors associated with market segments, and

$$G_j(\ldots) = \frac{\partial G_\gamma}{\partial y_j}(e^{V_{1n}}, \ldots, e^{V_{Jn}}). \tag{1.18}$$

We now obtain the probability model by integrating with respect to ξ and ω, that is $P(i|\mathcal{C}_n, X_n; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \lambda, \gamma) =$

$$\int_{\omega=0}^{1} \int_{\xi=-\infty}^{+\infty} P(i|\mathcal{C}_n, X_n; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \lambda, \gamma; \xi, \omega)\phi(\xi; 0, I)d\xi d\omega \tag{1.19}$$

where $\phi(\xi; 0, I)$ denotes the pdf of the multivariate normal density centered at zero with covariance matrix I, evaluated at ξ. If we gather all unknown parameters in a vector θ, the choice model (1.19) can be written

$$P(i|\mathcal{C}_n, X_n; \theta) \tag{1.20}$$

The generality of this model provides a great deal of modeling flexibility. Not only many models are special cases of this formulation, as described below, but new models can be derived in the same context. However, if more complex models are required, PythonBiogeme has to be used, whih does not assume any model structure. The model to be estimated has to be explicitly coded in Python language.

Internal note: The following mut be moved to the syntax section

We now link this formulation with the entries of the model file (Section 2.4). A normally distributed random parameter is mentioned in the definition of the utility function using the syntax

```
BETA [ GAMMA ]
```

For each such random parameter, there must be two entries in the Section [Beta], one corresponding to the entry in β_0 (`BETA` in the example above), and the other corresponding to the associated diagonal element of Γ (`GAMMA`). If Γ is diagonal, this is sufficient. If Γ is not diagonal, each non-zero off-diagonal entry must be listed in the Section [ParameterCovariances]. Each off-diagonal entry is

designed to capture the covariance between two random parameters. The name of a random parameter `BETA [ GAMMA ]` is by convention `BETA_GAMMA`.

Note that Biogeme 2.4 estimates the entries of β_0 and of Γ. But for most practical applications, the value of Γ is irrelevant, and the values within $\Gamma\Gamma^T$ are needed. The section `Variance of random coefficients` in the report files reports the entries of the $\Gamma\Gamma^T$ matrix. Note that if Γ is diagonal, those are simply the square of the estimated values reported in the `Utility parameters` section.

A uniformly distributed random parameter is mentioned in the definition of the utility function using the syntax

```
BETAA { BETAB }
```

For each such random parameter, there must be two entries in the Section [Beta], one corresponding to the entry in β_a (`BETA` in the example above), and the other corresponding to the entry in β_b(`BETB` in the example above).

1.2 Maximum likelihood estimation

We assume that we have access to a sample of N individuals, such that for each individual n, we have access to the choice set $\mathcal{C}_n$, the set of variables X_n (containing the socio-economic characteristics s_n and the attributes of each alternative z_{in}, $i \in \mathcal{C}_n$) as well as one observed choice i_n. Such a sample is usually referred to as *cross-sectional* data. Given a choice model (1.20), the likelihood of the sample is defined as the probability that the model correctly predicts all observed choices, that is

$$\mathcal{L}'(\theta) = \prod_{n=1}^{N} P(i_n|\mathcal{C}_n, X_n; \theta). \tag{1.21}$$

It is actually more convenient to refer to the log likelihood, that is

$$\mathcal{L}(\theta) = \log \mathcal{L}'(\theta) = \sum_{n=1}^{N} \log P(i_n|\mathcal{C}_n, X_n; \theta). \tag{1.22}$$

The maximum likelihood estimation of the parameters consists in finding the value of θ that maximizes (1.22). The main functionality of Biogeme 2.4 is to

solve the optimization problem

$$\max_{\theta} \mathcal{L}(\theta), \tag{1.23}$$

for a given choice model and a given sample file.

Estimation of the variance-covariance matrix

Under relatively general conditions, the asymptotic variance-covariance matrix of the maximum likelihood estimates is given by the Cramer-Rao bound

$$-\mathrm{E}\left[\nabla^2 \mathcal{L}(\theta)\right]^{-1} = \left\{-\mathrm{E}\left[\frac{\partial^2 \mathcal{L}(\theta)}{\partial\theta\partial\theta^{\mathsf{T}}}\right]\right\}^{-1}. \tag{1.24}$$

The term in square brackets is the matrix of the second derivatives of the log likelihood function with respect to the parameters evaluated at the true parameters. Thus the entry in the kth row and the ℓth column is

$$\frac{\partial^2 \mathcal{L}(\theta)}{\partial\theta_k \partial\theta_\ell}. \tag{1.25}$$

From the second order optimality conditions of the optimization problem, this matrix is negative definite if the maximum is unique, which is the algebraic equivalent of the local strict concavity of the log likelihood function.

Since we do not know the actual values of the parameters at which to evaluate the second derivatives, or the distribution of x_{in} and x_{jn} over which to take their expected value, we estimate the variance-covariance matrix by evaluating the second derivatives at the estimated parameters $\hat{\theta}$ and the sample distribution of x_{in} and x_{jn} instead of their true distribution. Thus we use

$$\mathrm{E}\left[\frac{\partial^2 \mathcal{L}(\theta)}{\partial\theta_k \partial\theta_\ell}\right] \approx \sum_{n=1}^{N} \left[\frac{\partial^2 \left(y_{in} \ln P_n(i) + y_{jn} \ln P_n(j)\right)}{\partial\theta_k \partial\theta_\ell}\right]_{\theta=\hat{\theta}}, \tag{1.26}$$

as a consistent estimator of the matrix of second derivatives. Denote this matrix as $\hat{A}$. Therefore, an estimate of the Cramer-Rao bound (1.24) is given by

$$\widehat{\Sigma}_{\theta}^{\mathrm{CR}} = -\hat{A}^{-1}. \tag{1.27}$$

If the matrix $\hat{A}$ is negative definite then $-\hat{A}$ is invertible and the Cramer-Rao bound is positive definite. However, this is not guaranteed.

Another consistent estimator of the (negative of the) second derivatives matrix can be obtained by the matrix of the cross-products of first derivatives as follows:

$$-\mathrm{E}\left[\frac{\partial^2 \mathcal{L}(\theta)}{\partial\theta\partial\theta^T}\right] \approx \sum_{n=1}^{N}\left(\frac{\partial \ell_n(\hat{\theta})}{\partial\theta}\right)\left(\frac{\partial \ell_n(\hat{\theta})}{\partial\theta}\right)^T = \hat{B}, \tag{1.28}$$

where

$$\left(\frac{\partial \ell_n(\hat{\theta})}{\partial\theta}\right) = \frac{\partial}{\partial\theta}(\log P(i_n|\mathcal{C}_n, X_n; \widehat{\theta})) \tag{1.29}$$

is the gradient vector of the likelihood of observation n. This approximation is employed by the BHHH algorithm, from the work by Berndt et al. (1974). Therefore, an estimate of the variance-covariance matrix is given by

$$\widehat{\Sigma}_\theta^{\mathrm{BHHH}} = \hat{B}^{-1}, \tag{1.30}$$

although it is rarely used. Instead, $\hat{B}$ is used to derive a third consistent estimator of the variance-covariance matrix of the parameters, defined as

$$\widehat{\Sigma}_\theta^{\mathrm{R}} = (-\hat{A})^{-1}\ \widehat{B}\ (-\hat{A})^{-1} = \widehat{\Sigma}_\theta^{\mathrm{CR}}\ (\widehat{\Sigma}_\theta^{\mathrm{BHHH}})^{-1}\ \widehat{\Sigma}_\theta^{\mathrm{CR}}. \tag{1.31}$$

It is called the *robust* estimator, or sometimes the *sandwich* estimator, due to the form of equation (1.31). Biogeme 2.4 reports statistics based on both the Cramer-Rao estimate (1.27) and the robust estimate (1.31).

When the true likelihood function is maximized, these estimators are asymptotically equivalent, and the Cramer-Rao bound should be preferred (Kauermann and Carroll, 2001). When other consistent estimators are used, the robust estimator must be used (White, 1982). Consistent non-maximum likelihood estimators, known as pseudo maximum likelihood estimators, are often used when the true likelihood function is unknown or difficult to compute. In such cases, it is often possible to obtain consistent estimators by maximizing an objective function based on a simplified probability distribution.

Panel data

Sometimes, it is possible to observe individuals over time. We assume that we have access to a sample of N individuals, such that for each individual n and for each time period $t = 1, \ldots, T$, we have access to the choice set $\mathcal{C}_{nt}$, the set of variables X_{nt} (containing the socio-economic characteristics s_n and the

attributes of each alternative z_{int}, $i \in \mathcal{C}_{nt}$) as well as one observed choice i_{nt}. Such a sample is usually referred to as *panel* data. In the presence of random parameters in the model, the likelihood function is slightly different if some of these parameters are distributed across individuals and not across observations. For the sake of simplicity here, we assume that all random parameters are distributed over individuals. In this case, the choice model for an individual n at time t, conditional to the value of ξ and ω, is given by (1.17)

$$P(i|\mathcal{C}_{nt}, X_{nt}; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \lambda, \gamma; \xi, \omega).$$

Now, for individual n, we observe the sequence of choices $i_{n1}, i_{n2}, \ldots, i_{nT}$. The likelihood of this sequence, again conditional to the value of ξ and ω, is given by

$$\prod_{t=1}^{T} P(i_{nt}|\mathcal{C}_{nt}, X_{nt}; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \lambda, \gamma; \xi, \omega). \tag{1.32}$$

We now need to integrate out the random parameters to obtain the contribution of individual n to the likelihood function:

$$\int_{\omega=0}^{1} \int_{\xi=-\infty}^{+\infty} \prod_{t=1}^{T} P(i_{nt}|\mathcal{C}_{nt}, X_{nt}; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \lambda, \gamma; \xi, \omega) \phi(\xi; 0, I) d\xi d\omega. \tag{1.33}$$

The log likelihood of the full sample in therefore

$$\mathcal{L} = \sum_{n=1}^{N} \log \int_{\omega=0}^{1} \int_{\xi=-\infty}^{+\infty} \prod_{t=1}^{T} P(i_{nt}|\mathcal{C}_{nt}, X_{nt}; \beta_f, \beta_0, \Gamma, \beta_a, \beta_b, \lambda, \gamma; \xi, \omega) \phi(\xi; 0, I) d\xi d\omega. \tag{1.34}$$

Choice based sampling

The estimation procedure described above is designed when the sample has be generated using an exogenous procedure, meaning that the probability for an individual in the population to be selected in the sample may depend on the independent (or exogenous) variables X_n, but not on the dependent variable (the choice). For this reason, it is usually called Exogenous Sample Maximum Likelihood (ESML). If the sample is choice-based, that is if the probability to be selected in the sample depends on the choice made, that estimator is not consistent anymore.

Bierlaire et al. (2008) have proposed an estimator for choice-based sample and MEV models. It consists in estimating new parameters, playing a role similar to alternative specific constants, but designed to absorb the bias due to choice-based samples. Ideally, the value of these parameters should be derived from the sampling strategy. However, it is possible to estimate them from data, although it is not necessarily recommended.

Chapter 2

BisonBiogeme

BisonBiogeme is the first version of Biogeme. It was first released in 2000, and has been improved permanently ever since. The idea is that the software provides a large family of choice models that are pre-implemented. The user has to specify the utility function, and various additional aspects of the model specification using a simple modeling language specifically designed for that purpose.

2.1 Walkthrough

In order to introduce the syntax of BisonBiogeme, we are explaining in details an example where a logit model with 3 alternatives is estimated. We use the Swissmetro example (see Bierlaire et al., 2001). The following files are necessary to run the example. They are available from `biogeme.epfl.ch`.

- The model specification file: `01logit.mod`
- The data file: `swissmetro.dat`

The model

The model is a logit model with 3 alternatives. The utility functions are defined as:

```
V_1 = V_TRAIN =  ASC_TRAIN + B_TIME * TRAIN_TT_SCALED
                           + B_COST * TRAIN_COST_SCALED
V_2 = V_SM = ASC_SM + B_TIME * SM_TT_SCALED
```

```
                        + B_COST * SM_COST_SCALED
V_3 = V_CAR =  ASC_CAR + B_TIME * CAR_TT_SCALED
                       + B_COST * CAR_CO_SCALED
```

where `TRAIN_TT_SCALED`, `TRAIN_COST_SCALED`, `SM_TT_SCALED`, `SM_COST_SCALED`, `CAR_TT_SCALED`, `CAR_CO_SCALED` are variables, and `ASC_TRAIN`, `ASC_SM`, `ASC_CAR`, `B_TIME`, `B_COST` are parameters to be estimated. Note that it is not possible to identify all alternative specific constants `ASC_TRAIN`, `ASC_SM`, `ASC_CAR` from data. Consequently, `ASC_SM` is normalized to 0.

The availability of an alternative `i` is determined by the variable `avi`, `i`=1,...3, which is equal to 1 if the alternative is available, 0 otherwise. The probability of choosing an available alternative `i` is given by the logit model:

$$P(i|\mathcal{C}) = \frac{e^{V_i}}{av1e^{V_1} + av2e^{V_2} + av3e^{V_3}} \tag{2.1}$$

Given a data set of N observations, the loglikelihood of the sample is

$$\mathcal{L} = \sum_n \log P(i_n|\mathcal{C}) \tag{2.2}$$

where i_n is the alternative actually chosen by individual n.

The data file

Biogeme assumes that the data file contains in its first line a list of labels corresponding to the available data, and that each subsequent line contains the exact same number of numerical data, each row corresponding to an observation. Delimiters can be tabs or spaces.

The data file used for this example is `swissmetro.dat`.

The model specification file

We explain here line by line the model specification file `01logit.mod`. It is organized into sections. In principle, the order in which the sections appear is irrelevant.

[ModelDescription]

This section allows to mention a description of the model that will be copied in the report file. Each line of the description must be delimited by double quotes.

```
[ModelDescription]
"Example of a logit model for a transportation mode choice with 3 alternatives:"
"- Train"
"- Car"
"- Swissmetro, an hypothetical high-speed train"
```

[Choice]

It simply describes to Biogeme where the dependent variable (that is, the chosen alternative) can be found in the file.

```
[Choice]
CHOICE
```

Note that the syntax is case sensitive, and that `CHOICE` is different from `choice`, and from `Choice`.

Beta]

Each parameter to be estimated must be declared in this section. For each parameter, the following must be mentioned:

1. the name of the parameter
2. the default value
3. a lower bound
4. an upper bound
5. a flag that indicates if the parameter must be estimated (0) or if it keeps its default value (1).

```
[Beta]
// Name Value  LowerBound UpperBound  status (0=variable, 1=fixed)
ASC_CAR  0 -10              10              0
ASC_TRAIN   0 -10              10              0
ASC_SM         0 -10              10              1
B_TIME 0 -10              10              0
B_COST 0 -10              10              0
```

Note that the fifth entry for `ASC_SM` is 1, as we want to keep it to its default value, that is 0.0.

[LaTeX]

Among other output files, Biogeme generates a file in LaTeX format. In this section, the name of the parameters can be specified in LaTeX syntax, to appear properly in the output file.

```
[LaTeX]
ASC_CAR "Cte. car"
ASC_SBB "Cte. train"
ASC_SM "Cte. Swissmetro"
B_TIME "$\beta_\text{time}$"
B_COST "$\beta_\text{cost}$"
```

[Utilities]

The specification of the utility functions is described in this section. The specification for one alternative must start at a new row, and may actually span several rows. For each of them, four entries are specified:

1. The identifier of the alternative, with a numbering convention consistent with the section `[Choice]`.
2. The name of the alternative.
3. The availability condition. In this case, it is a direct reference to one of the entries in the data file. The convention is that zero is treated as "false", and one is treated as "true". Actually, any value different from zero is considered as "true".
4. The linear-in-parameter utility function is composed of a list of terms, separated by a +. Each term is composed of the name of a parameter and the name of an attribute, separated by a $\star$. Note that a space is required after each parameter name.

```
[Utilities]
// Id Name  Avail  linear-in-parameter expression
    1 A1_TRAIN TRAIN_AV_SP ASC_TRAIN * one
                           + B_TIME * TRAIN_TT_SCALED
                           + B_COST * TRAIN_COST_SCALED
```

```
2 A2_SM    SM_AV          ASC_SM * one
                        + B_TIME * SM_TT_SCALED
                        + B_COST * SM_COST_SCALED
3 A3_Car   CAR_AV_SP     ASC_CAR * one
                        + B_TIME * CAR_TT_SCALED
                        + B_COST * CAR_CO_SCALED
```

[Expressions]

It describes to Biogeme how to compute attributes not directly available from the data file.

- When boolean variables are involved, the value TRUE is represented by 1, and the value FALSE is represented by 0. Therefore, a multiplication involving a boolean variable is equivalent to a ”AND” operator.

```
CAR_AV_SP =  CAR_AV   * (  SP   !=  0  )
TRAIN_AV_SP =  TRAIN_AV   * (  SP   !=  0  )
SM_COST =  SM_CO   * (  GA   ==  0  )
TRAIN_COST =  TRAIN_CO   * (  GA   ==  0  )
```

- Variables can be rescaled

```
TRAIN_TT_SCALED = TRAIN_TT / 100.0
TRAIN_COST_SCALED = TRAIN_COST / 100
SM_TT_SCALED = SM_TT / 100.0
SM_COST_SCALED = SM_COST / 100
CAR_TT_SCALED = CAR_TT / 100
CAR_CO_SCALED = CAR_CO / 100
```

[Exclude]

It contains a boolean expression that is evaluated for each observation of the data file. Each observation such that this expression is “true” is discarded from the sample. Here, the modeler has developed the model only for work trips. Observations such that the dependent variable CHOICE is 0 are also removed.

```
(( PURPOSE != 1 ) * (  PURPOSE   !=  3  ) + ( CHOICE == 0 ))
```

[Model]

It tells Biogeme which assumptions must be used regarding the error term, that is which type of model must be estimated. In this example, it is the logit model (or MNL, for *multinomial logit*, as it is sometimes called).

```
[Model]
// $MNL stands for MultiNomial Logit
$MNL
```

Running biogeme

If Biogeme has been installed properly, the estimation is started with the following statement:

```
biogeme 01logit swissmetro.dat
```

The following appears on the screen:

- Information about the version of Biogeme. The date is when the software was compiled.

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
biogeme 2.4 [Mer 9 jul 2014 15:27:09 CEST]
Michel Bierlaire, EPFL
-- Compiled by michelbierlaire on Darwin
See http://biogeme.epfl.ch
                    !! CFSQP is available !!
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
"In every non-trivial program there is at least one bug."
```

- Biogeme checks if a file called `mymodel.par`, containing various parameters, exists. If not, it checks if the file called `default.par` exists. If not, it creates it and set default values to the parameters. That's what most users need in the beginning. Note that the information like `[15:48:50]patFileNames.cc:49` can be safely ignored.

```
[14:57:01]patFileNames.cc:49  01logit.par does not exist
[14:57:01]patFileNames.cc:53  Trying default.par instead
[14:57:01]patBiogeme.cc:178  File default.par does not exist. Default value
[14:57:01]patBiogeme.cc:180  A file default.par has been created
```

- Biogeme then reads the model and data files and reports various information.

```
 Opening file swissmetro.dat
 Data  file... line 500 Memory: 97 Kb
 Data  file... line 1000 Memory: 184 Kb
 Data  file... line 1500 Memory: 184 Kb
 Data  file... line 2000 Memory: 191 Kb
```

```
Data  file... line 2500 Memory: 289 Kb
Data  file... line 3000 Memory: 386 Kb
Data  file... line 3500 Memory: 484 Kb
Data  file... line 4000 Memory: 503 Kb
Data  file... line 4500 Memory: 600 Kb
Data  file... line 5000 Memory: 647 Kb
Data  file... line 5500 Memory: 745 Kb
Data  file... line 6000 Memory: 842 Kb
Data  file... line 6500 Memory: 940 Kb
Data  file... line 7000 Memory: 1 Mb
Data  file... line 7500 Memory: 1 Mb
Data  file... line 8000 Memory: 1 Mb
Data  file... line 8500 Memory: 1 Mb
Data  file... line 9000 Memory: 1 Mb
Data  file... line 9500 Memory: 1 Mb
Data  file... line 10000 Memory: 1 Mb
Data  file... line 10500 Memory: 1 Mb
Total obs.:   10727
Total memory: 1321.88 Kb
Run time for data processing: 00:01
```

- Biogeme then starts the estimation. It displays miscellaneous information at each iteration of the estimation algorithm.

```
 Init loglike=-6964.66
    gmax Iter   radius        f(x)     Status       rhok nFree
+1.44e-03    1 1.00e+00 +6.9646630e+03 ****Converg  +1.05e+00 4  ++
+1.82e-03    2 2.00e+00 +5.5911931e+03 ****Converg  +9.74e-01 4  ++
+1.93e-03    3 4.00e+00 +5.3677413e+03 ****Converg  +1.12e+00 4  ++
+2.04e-03    4 8.00e+00 +5.3424604e+03 ****Converg  +1.52e+00 4  ++
+1.92e-03    5 1.60e+01 +5.3362826e+03 ****Converg  +1.66e+00 4  ++
+1.92e-03    6 3.20e+01 +5.3336219e+03 ****Converg  +1.69e+00 4  ++
+1.97e-03    7 6.40e+01 +5.3324003e+03 ****Converg  +1.69e+00 4  ++
+2.01e-03    8 1.28e+02 +5.3318102e+03 ****Converg  +1.70e+00 4  ++
+2.03e-03    9 2.56e+02 +5.3315246e+03 ****Converg  +1.70e+00 4  ++
+2.03e-03   10 5.12e+02 +5.3313855e+03 ****Converg  +1.70e+00 4  ++
+1.43e-03   11 1.02e+03 +5.3313175e+03 ****Converg  +1.70e+00 4  ++
+1.01e-03   12 2.05e+03 +5.3312842e+03 ****Converg  +1.70e+00 4  ++
+7.11e-04   13 4.10e+03 +5.3312679e+03 ****Converg  +1.70e+00 4  ++
+5.00e-04   14 8.19e+03 +5.3312598e+03 ****Converg  +1.70e+00 4  ++
+3.52e-04   15 1.64e+04 +5.3312559e+03 ****Converg  +1.70e+00 4  ++
+2.47e-04   16 3.28e+04 +5.3312539e+03 ****Converg  +1.70e+00 4  ++
+1.74e-04   17 6.55e+04 +5.3312529e+03 ****Converg  +1.70e+00 4  ++
+1.22e-04   18 1.31e+05 +5.3312525e+03 ****Converg  +1.70e+00 4  ++
+8.58e-05   19 2.62e+05 +5.3312522e+03 ****Converg  +1.70e+00 4  ++
```

```
 +6.03e-05   20 5.24e+05 +5.3312521e+03 ****Converg  +1.70e+00 4  ++
 +4.23e-05   21 1.05e+06 +5.3312521e+03 ****Converg  +1.70e+00 4  ++
 +2.97e-05   22 2.10e+06 +5.3312520e+03 ****Converg  +1.70e+00 4  ++
 +2.09e-05   23 4.19e+06 +5.3312520e+03 ****Converg  +1.70e+00 4  ++
 +1.47e-05   24 8.39e+06 +5.3312520e+03 ****Converg  +1.70e+00 4  ++
 +1.03e-05   25 1.68e+07 +5.3312520e+03 ****Converg  +1.70e+00 4  ++
 +7.24e-06   26 3.36e+07 +5.3312520e+03 ****Converg  +1.70e+00 4  ++

 Convergence reached...
--> time interval [14:57:02,14:57:03]
```

- Biogeme reports the running time and prepares the output files.

```
Run time: 00:01
 Final log-likelihood=-5331.25
 Be patient... BIOGEME is preparing the output files
--> time interval [14:57:03,14:57:03]
 Run time for var/covar computation: 00:00
```

- For the record, Biogeme reports the list of files that were actually used as input.

```
 BIOGEME Input files
 ===================
 Parameters: default.par
 Model specification: 01logit.mod
 Sample 1 : swissmetro.dat
```

- Biogeme reports the list of files that have been created, containing the results of the estimation, as well as many other pieces of information.

```
 BIOGEME Output files
 ====================
 Estimation results: 01logit.rep
 Estimation results (HTML): 01logit.html
 Estimation results (Latex): 01logit.tex
 Estimation results (ALogit): 01logit.F12
 Result model spec. file: 01logit.res
 Sample statistics: 01logit.sta
```

- Biogeme reports also the name of files that may be helpful in understanding problems with the model.

```
BIOGEME Debug files
===================
Log file: 01logit.log
Parameters debug: parameters.out
Model debug: model.debug
Model spec. file debug: __specFile.debug
```

- Biogeme reports some information specific to the model. For logit, it reports the minimum argument of all exponentials computed during the process, in order to signal a possible underflow. Most users do not worry about this information.

```
Model informations: Multinomial Logit Model
==================
The minimum argument of exp was -18.352

Run time for estimation:      00:01
Total run time:               00:02
```

- For the results, most users will consult the HTML file `01logit.html` with their preferred browser. A file written in ASCII format is also available, with the extension `.rep`. A file with LaTeX code is also created, so that the results can easily be integrated in a report or an article written with this word processor.¡/p¿

2.2 Invoking Biogeme 2.4

Biogeme 2.4 is invoked in a shell under Linux, in a DOS command window or a Cygwin command window under Windows using the following statement structure

```
biogeme model_name sample_file_1 sample_file2 sample_file3 ...
```

By default, the `sample_file_1` is assumed to be `sample.dat`, and the `model_name` to be `default`. Therefore, typing

```
biogeme model_name
```

is equivalent to typing

```
biogeme model_name sample.dat
```

and typing

```
biogeme
```

is equivalent to typing

```
biogeme default sample.dat
```

Finally, typing

```
biogeme -h
```

generates an output looking like

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
BIOGEME Version n.x [date]
Michel Bierlaire, EPFL
-- Compiled by Michel Bierlaire on MINGW32_NT-5.1
See http://biogeme.epfl.ch
                    !! CFSQP is available !!
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        "In every non-trivial program there is at least one bug."

Usage: biogeme model_name sampleFile1 sampleFile2 sampleFile3 ...
```

If the name of the model is `mymodel`, say, Biogeme 2.4 reads the following files:

- a file containing the parameters controlling the behavior of Biogeme 2.4: `mymodel.par` (Section 2.3)
- a file containing the model specification: `mymodel.mod` (Section 2.4)
- a file containing the data: `sample.dat` (Section 2.5)
- optionally a file containing the random numbers to use if estimation is based on simulation.

It automatically generates the following output files:

- a file reporting the results of the estimation: `mymodel.rep` (section 2.8),
- the same file in HTML format,

- a file containing the main results in LaTeX format: `mymodel.tex`,
- a file containing the main results in ALogit format: `mymodel.F12`,
- a file containing the specification of the estimated model, in the same format as the model specification file: `mymodel.res`
- a file containing the specification of the estimated model at each iteration, in the same format as the model specification file: `mymodel.bck` (only if the parameter `gevSaveIntermediateResults` is set to one),
- a file containing some descriptive statistics on the data: `mymodel.sta` (section 2.7),

and the following files to help understanding possible problems

- a file containing messages produced by Biogeme 2.4 during the run: `mymodel.log`
- a file containing the values of the parameters which have been actually used by Biogeme 2.4: `parameters.out`
- a file containing the data stored in Biogeme 2.4 to represent the model: `model.debug`
- a file containing the specification of the model, as it has actually been understood by Biogeme 2.4: `__specFile.debug`

These file names may be modified, according to the following rules:

1. If an input file `mymodel.xxx` does not exist, Biogeme 2.4 attempts to open the file `default.xxx`. If this file does not exist, Biogeme 2.4 exits with an error. Typically, the parameter file is not model-dependent. Therefore, it is common to call it `default.par` to avoid copying it for each different model to be estimated.
2. If an output file `mymodel.xxx` already exists, Biogeme 2.4 does not overwrite it. Instead, it creates the file `mymodel~1.xxx`. If the file `mymodel~1.xxx` exists, Biogeme 2.4 creates the file `mymodel~2.xxx`, and so on.

To avoid any ambiguity, Biogeme 2.4 displays the filenames it has actually used for a specific run, for instance

```
BIOGEME Input files
===================
Parameters:                      default.par
Model specification:             mymodel.mod
Sample 1 :                       sample.dat
Sample 2 :                       sample2.dat
BIOGEME Output files
====================
Estimation results:              mymodel~3.rep
Estimation results (HTML):       mymodel~3.html
Estimation results (Latex):      mymodel~5.tex
Estimation results (ALogit):     mymodel~1.F12
Result model spec. file:         mymodel~2.res
Sample statistics:               mymodel~1.sta
BIOGEME Debug files
===================
Log file:                        mymodel.log
Parameters debug:                parameters.out
Model debug:                     model.debug
Model spec. file debug:          __specFile.debug
```

Biogeme 2.4 also generates a file called `summary.html` where a summary of all runs performed in the working directory are gathered. The name of this file can be modified (Section 2.3).

It is highly recommended to regularly clean the working directory and save the output files in a different place.

Graphical user's interface

The version of Biogeme with a Graphical user's interface (GUI) is a very simple interface (see Figures 2.1 on the facing page and 2.2 on the next page) developed with the library Fast Light Toolkit.

The user must select the model specification file and the data file, then click one of the two buttons:

- `Estimate` for Biogeme.
- `Simulate` for Biosim.

If the run completes successfully, the name of the report file is displayed at the bottom of the screen. It can be viewed by clicking on the appropriate button.

Figure 2.1: Graphical User Interface when `winbiogeme.exe` is launched

Figure 2.2: Graphical User Interface when the run is finished

2.3 Parameter file

The parameter file provides the parameters controlling the execution of Biogeme 2.4. It is not mandatory. If it does not exist, Biogeme 2.4 uses the default values, and automatically creates a file named `default.par`. If entries are missing in the file, Biogeme 2.4 will use the default values.

The file is divided into sections, each section containing a list of parameters and their corresponding value.

Section [GEV]

The five first parameters are the only parameters which most users will ever use. The others are sorted alphabetically.

`gevAlgo` It selects the optimization algorithm to be used for log-likelihood estimation. As of now, `"BIO"`, `"BIOMC"`, `"CFSQP"`, `"SOLVOPT"` and `"DONLP2"` are valid entries. The default is `"BIO"`. More details about these algorithms are available at Section 2.11.

`gevScreenPrintLevel` This parameter defines the level of display to be produced on the screen during a run. Valid values are 1 for general messages only, 2 for detailed messages, and 3 for debug messages. Default: 1.

`gevLogFilePrintLevel` This parameter defines the level of display to be produced in the log file during a run. Valid values are 1 for general messages only, 2 for detailed messages, and 3 for debug messages. Default: 2.

`gevPrintVarCovarAsList` If set to 1, the variance-covariance matrix of the estimated parameters is displayed as a list (one row per entry). Default: 1.

gevPrintVarCovarAsMatrix If set to 1, the variance-covariance matrix of the estimated parameters is displayed as a matrix. We recommend to use this feature only if the number of parameters is small (not more than 10). Default: 0.

gevAutomaticScalingOfLinearUtility If 1, linear utility functions are automatically scaled to avoid numerical problems during the estimation. The scaling is computed in such a way that all attributes have a level of magnitude of about 1.0. Default value: 0.

gevBinaryDataFile This is the name of the binary data file where the processed data are stored. Default: `__BiogemeData.bin`.

gevBufferSize Biogeme 2.4 reads the first line of the data files, and stores it in a buffer to analyze it and extract the labels. The size of the buffer is determined by this parameter. The default value is 100'000. Adapt the value if the first line of your data file contains more that 99'999 characters. BIOGEME provides a warning if the default value is exceeded.

gevCheckDerivatives If set to 1, the analytical derivatives of the log-likelihood functions and the nonlinear constraints are compared to the finite difference derivatives. This is used basically when a new model is included and for debugging purposes. Default value: 0.

gevDataFileDisplayStep While pre-processing the data file before the estimation, Biogeme 2.4 reports progress each time it has read a given number of rows. This number is specified by the parameter `getDataFileDisplayStep`, and its default value is 500.

gevDebugDataFirstRow Biogeme 2.4 can print what it actually reads from the data file. This parameter is the number of the first row for which his information is displayed. It is recommended to use it when strange results are generated by the package. It helps identifying garbage in the data file, such as strings, for instance. Default: 0.

gevDebugDataLastRow Biogeme 2.4 can print what it actually reads from the data file. This parameter is the number of the last row for which this information is displayed. Default: 0.

gevDecimalDigitsStats Number of digits after the decimal points to be used for printing general statistics in the output files. Default: 3.

gevDecimalDigitsTTest Number of digits after the decimal points to be used for printing t-tests in the output files. Default: 2.

gevDumpDrawsOnFile If set to 1, Biogeme 2.4 dumps the draws used for simulated likelihood estimation. The name of the file is displayed at the end of the run. If the model name is `model`, the filename is `model.draws`. Default value: 0.

gevForceScientificNotation If 1, use the scientific notation for printing results, like in previous versions of Biogeme. Default: 0.

gevGenerateActualSample If set to 1, Biogeme 2.4 generates a copy of the sample file containing only the observations that have not been excluded. Default: 0.

gevMinimumMu When the homogeneity parameter μ of GEV models is estimated, its theoretical lower bound must be zero. However, numerically, a value of 0 generates problems during the computation of the model. Therefore, the lower bound is automatically set to the value defined by this parameter. Default: 1.0e-5.

gevMaxPrimeNumber The generation of Halton sequences is based on prime numbers. This parameter defines the maximum number of prime numbers that can be used. Most users will never have to change the default value. But if it is too low, an error message is generated:

```
Warning: Error: 23 Halton series must be
  generated, but there are only 10 prime
 numbers available. Increase the value of
 gevMaxPrimeNumber in the parameters file
```

Default value: 1000.

gevMissingValue This parameter is used mainly for debugging purposes. It defines the value given to missing values in the data file. If one of them is used in the computation of the utility functions, an error message is triggered. Default value: 99999.0

gevOutputActualSample If parameter `gevGenerateActualSample` is set to 1, this parameter defines the name of the file where the sample is saved. Default: `__actualSample.dat`.

gevPrintPValue If 1, print the p-value in the results. The p-value is computed as follows: if t is the t-test of the parameters,

$$p = 2(1 - \Phi(t)), \tag{2.3}$$

where $\Phi(\cdot)$ is the cumulative density function of the univariate normal distribution. Default: 1.

gevRandomDistrib There are three valid entries for this parameter: `PSEUDO`, `MLHS` and `HALTON`. If `PSEUDO` is selected, maximum simulated likelihood is based on pseudo-random draws. If `HALTON` is selected, Halton sequences are generated. If `MLHS` is selected, a Modified Latin Hypercube Sampling strategy is adopted (see Hess et al., 2006). Default: `PSEUDO`

gevSaveIntermediateResults If 1, the current estimates are saved at each iteration in a file with extension `.bck`. This is particularly useful for models that take a while to estimate, so that the estimation can be restarted from the last iterate. Default: 0.

gevSeed It defines the seed value for the pseudo-random number generator. Default value: 9021967

gevSignificantDigitsParameters Number of significant digits to be used for printing estimated parameters in the output files. Default: 3.

gevSingularValueThreshold Identification problems are analyzed using a Singular Value Decomposition procedure. If a singular value is small (that is, its absolute value is less than the value defined by this parameter), the model is considered degenerate and the source of this degeneracy is displayed. Default: 1.0e-4.

gevStopFileName During the optimization process, Biogeme 2.4 checks for the existence of a file, whose name is defined by this parameter. If the file exists, Biogeme 2.4 interrupts the iterations and generate output files. This is convenient to prematurely stop iterations without loosing the computations performed thus far. The default value is `"STOP"`.

gevStoreDataOnFile Biogeme 2.4 uses a database gathering the processed data from the file provided by the user and, if applicable, the draws for the simulated maximum likelihood estimation. If the parameter is 0, the database is stored in memory. If 1, it is stored in the

binary file defined by the parameter gevBinaryDataFile. It is recommended to use 0, except if the data does not fit in memory. Indeed, accessing to the file slows down the estimation process. Default: 0.

gevSummaryFile Name of the file summarizing several runs of Biogeme 2.4. Default value: `summary.html`

gevSummaryParameters Name of the file containing the name of the parameters whose estimated values must be reported in the summary file. Default value: `summary.lis`

gevVarCovarFromBHHH The computation of the variance-covariance matrix of the estimated parameters using finite difference approximation may take a while for complex models. It is sometimes useful to use the BHHH approximation, which is much faster to compute. If so, set this parameter to 1. It is recommended not to use BHHH in the final model. Default: 0.

gevTtestThreshold Set the threshold for the t-test hypothesis tests. If the absolute value of a t-test is less than `gevTtestThreshold`, a symbol $\star$ will be appended to the relevant line in the report file (Section 2.8). Default value: 1.96.

gevWarningLowDraws Biogeme 2.4 displays a warning if the number of draws for simulated maximum likelihood estimation is considered too low. This parameter defines the threshold used in the generation of this warning message. Note that it has no effect on the estimation itself. Default: 1000.

gevWarningSign When a t-test is not successful, a warning size is displayed in the report file and in the HTML file. This parameter defines the nature of this sign. Default value: $\star$.

The following are new in Biogeme 2.4:

gevNumberOfThreads When Biogeme 2.4 is compiled to work with parallel processors, this parameter specifies the number of threads that will be launched. Note that it may exceed the actual number of available processors. However, this may affect the performance by creating unnecessary overhead. It is therefore advised to set this parameter to the exact number of available processors. Default: 4.

gevOne Name of the expression that is replaced by the value 1.0. It can be used in the specification of the utility without explicitly defining it in the Section [Expressions]. Default: `one`.

gevEigenvalueThreshold An eigenvalue is considered to be zero (and the matrix considered to be singular) if its absolute value is less or equal to the value of this parameter. Default: 1.0e-6.

The following are new in Biosim 2.4:

gevNonParamPlotRes This parameter defines the number of equally distributed values on the x-axis used to generate nonparametric plots. Default: 100.

gevNonParamPlotMaxY When generating nonparametric plots, values larger that this parameter are considered equal to the parameter. Symmetrically, values lower than the negative parameter are considered equal to the negative value. Default: 1000.0.

gevNonParamPlotXSizeCm Width in centimeters of the nonparametric plots in the LaTeX output. Default: 15.

gevNonParamPlotYSizeCm Height in centimeters of the nonparametric plots in the LaTeX output. Default: 10.

gevNonParamPlotMinXSizeCm Units on the x-axis are computed automatically for nonparametric plots, but will no be lower than the value of this parameter. Default: 0.00001.

gevNonParamPlotMinYSizeCm Units on the y-axis are computed automatically for nonparametric plots, but will no be lower than the value of this parameter. Default: 0.00001.

Section [BasicTrustRegion]

This section is designed for the BIO and BIOMC optimization algorithms (see Section 2.11).

BTRMaxIter Maximum number of iterations to be performed. Default: 1000.

BTRTypf Typical value of the log-likelihood function (see Section 2.11). Default: 1.0.

BTRTolerance Value used for the stopping criterion (see Section 2.11). Default: 6.05545e-06.

BTRCheapHessian If 1, BHHH (see Berndt et al., 1974) is used as an approximation of the second derivatives matrix. Default: 1.

BTRUsePreconditioner If 1, the subproblem is preconditioned using a modified Cholesky factorization (**?**). Default: 0

BTRInitRadius Defines the initial radius of the trust region. Default: 1.

BTRIncreaseTRRadius Defines the factor by which the trust region is updated. Default: 2.

BTRMinTRRadius Defines the minimum radius of the trust region. If this radius is reached, the iterations are interrupted. Default: 1.0e-7.

BTRMaxTRRadius Defines the maximum radius of the trust region. If this radius is reached, the trust region is not enlarged anymore. Default: 1.0e10.

BTRStartDraws If BIOMC is used for simulated maximum likelihood estimation, this parameter defines the number of draws which are used during the first iterations. Default: 10.

BTRIncreaseDraws If BIOMC is used for simulated maximum likelihood estimation, this parameters defines the factor by which the number of draws is increased. Default: 2.

Section [cfsqp] This section is designed to define parameters needed by the CFSQP algorithm (Section 2.11).

cfsqpIprint Set it to 1 for silent mode, and to 2 for information at each iteration of the optimization algorithm. Default is 1.

cfsqpMaxIter Maximum number of iterations. Default is 500.

cfsqpMode Even if it is a descent algorithm, CFSQP sometimes allows non-monotone iterates, hoping not to be trapped in local minima. If the function is convex, a descent algorithm is more appropriate. In this case, set the value to 100. See CFSQP manual for more details. Default is 110.

cfsqpEps See CFSQP manual. Default is 6.05545e-06. In general, it should not be changed.

cfsqpEpsEqn See CFSQP manual. Default is 6.05545e-06. In general, it should not be changed.

cfsqpUdelta See CFSQP manual. Default is 0.0. In general, it should not be changed.

Section [solvopt] This section is designed to define parameters needed by the SOLVOPT algorithm (Section 2.11).

solvoptMaxIter Maximum number of iterations. Default is 15000.

solvoptDisplay Controls the display of the algorithm. See SOLVOPT manual. Default is 1.

solvoptErrorArgument See SOLVOPT manual. Default is 1.0e-4. In general, it should not be changed.

solvoptErrorFunction See SOLVOPT manual. Default is 1.0e-6. In general, it should not be changed.

Section [donlp2] This section is designed to define parameters needed by the DONLP2 algorithm (Section 2.11).

donlp2Epsx See DONLP2 manual. Default is 1.0e-5. In general, it should not be changed.

donlp2Delmin See DONLP2 manual. Default is 1.0e-6. In general, it should not be changed.

donlp2Smallw See DONLP2 manual. Default is 3.66685e-11. In general, it should not be changed.

donlp2Epsdif See DONLP2 manual. Default is 0.0. In general, it should not be changed.

donlp2NReset See DONLP2 manual. Default is 9 . In general, it should not be changed.

Internal note: Check the status of the comment below.

It seems that syntax errors in `default.par` cause Biogeme 2.4 to skip the rest of the file, ignoring all remaining parameters without complaining. This "bug" still has to be fixed. Biogeme 2.4 writes in the file `parameters.out` the values of the parameters that have been actually used. Make sure you check this file regularly.

Name	Value
gevAlgo	BIO
gevScreenPrintLevel	1
gevLogFilePrintLevel	2
gevPrintVarCovarAsList	1
gevPrintVarCovarAsMatrix	0
gevAutomaticScalingOfLinearUtility	0
gevBinaryDataFile	__BiogemeData.bin
gevBufferSize	100000
gevCheckDerivatives	0
gevDataFileDisplayStep	500
gevDebugDataFirstRow	0
gevDebugDataLastRow	0
gevDecimalDigitsStats	3
gevDecimalDigitsTTest	2
gevDumpDrawsOnFile	0
gevEigenvalueThreshold	1.0e-6
gevForceScientificNotation	0
gevGenerateActualSample	0
gevMinimumMu	1.0e-5
gevMaxPrimeNumber	1000
gevMissingValue	99999

Table 2.1: Default values of the parameters

2.4 Model specification file

The file `mymodel.mod` contains the specification of the discrete choice model to be estimated. The sections of this file have to be specified as described below. Note that comments can be included using `//`. All characters after this command, up to the end of the current line, are ignored.

`[ModelDescription]` Type here any text that describes the model. It may contain several lines. Each line must be within double-quotes, like this

```
[ModelDescription]
"This is the first line of the model description"
"This is the second line of the model description"
```

Name	Value
gevNonParamPlotMaxY	1000.0
gevNonParamPlotMinXSizeCm	0.00001
gevNonParamPlotMinYSizeCm	0.00001
gevNonParamPlotRes	100
gevNonParamPlotXSizeCm	15
gevNonParamPlotYSizeCm	10
gevNumberOfThreads	4
gevOne	one
gevOutputActualSample	__actualSample.dat
gevPrintPValue	1
gevRandomDistrib	PSEUDO
gevSaveIntermediateResults	0
gevSeed	9021967
gevSignificantDigitsParameters	3
gevSingularValueThreshold	1.0e-4
gevStopFileName	STOP
gevStoreDataOnFile	0
gevSummaryFile	summary.log
gevSummaryParameters	summary.lis
gevVarCovarFromBHHH	0
gevTtestThreshold	1.96
gevWarningLowDraws	1000
gevWarningSign	*

Table 2.2: Default values of the parameters (ctd)

Note that it will be copied verbatim in the LaTeX file. Therefore, if it contains special characters which are interpreted by LaTeX, such as $ or &, you may need to edit the LaTeX file before processing it.

[Choice] Provide here the formula to compute the identifier of the chosen alternative from the data file. Typically, a "`choice`" entry will be available directly in the file, but any formula can be used to compute it. Assume for example that you have numbered alternatives 100, 200 and 300. But in the data file, they are numbered 1,2 and 3. In this case, you must write

```
[Choice]
 100  *  choice
```

Any expression described in Section [Expressions] is valid here.

[Weight] Provide here the formula to compute the weight associated to each observation. The weight of an observation will be multiplied to the corresponding term in the log-likelihood function. Ideally, the sum of the weights should be equal to the total number of observations, although it is not required. The file reporting the statistics contains a recommendation to adjust the weights in order to comply with this convention.

Internal note: Check what happens if biosim is called with weights

Important: do not use the weight section in Biosim 2.4.

[Beta] Each line of this section corresponds to a parameter of the utility functions. Five entries must be provided for each parameter:

1. Name: the first character must be a letter (any case) or an underscore (_), followed by a sequence of letters, digits, underscore (_) or dashes (−), and terminated by a white space. Note that case sensitivity is enforced. Therefore `varname` and `Varname` would represent two different variables.
2. Default value that will be used as a starting point for the estimation, or used directly for the simulation in BIOSIM.

3. Lower bound on the valid values[1];
4. Upper bound on the valid values;
5. Status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given default value.

Note that this section is independent of the specific model to be estimated, as it captures the deterministic part of the utility function.

```
[Beta]
// Name  Value      LowerBound  UpperBound status
   ASC1  0          -10000      10000      1
   ASC2  -0.159016  -10000      10000      0
   ASC3  -0.0869287 -10000      10000      0
   ASC4  -0.51122   -10000      10000      0
   ASC5  0.718513   -10000      10000      0
   ASC6  -1.39177   -10000      10000      0
   BETA1 0.778982   -10000      10000      0
   BETA2 0.809772   -10000      10000      0
```

[Mu] μ is the homogeneity parameter of the MEV model. Usually, it is constrained to be one. However, Biogeme 2.4 enables to estimate it if requested (see example `10nl-bottom.mod` for a nested logit model normalized from the bottom, so that μ is estimated). Four entries are specified here:

1. Default value that will be used as a starting point for the estimation (common value: 1.0);
2. Lower bound on the valid values (common value: 1.0e-5);
3. Upper bound on the valid values (common value: 1.0);
4. Status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given value.

[Utilities] Each row of this section corresponds to an alternative. Four entries are specified:

[1] Bounds specification is mandatory in Biogeme 2.4. If you do not want bounds, just put large negative values for lower bounds and large positive values for upper bounds. Anyway, if the bound is not active at the solution, it does not play any role, except for safeguarding the algorithm.

1. The identifier of the alternative, with a numbering convention consistent with the choice definition;
2. The name of the alternative: the first character must be a letter (any case) or an underscore (_), followed by a sequence of letters, digits, underscore (_) or dashes (−), and terminated by a white space;
3. The availability condition: this must be a direct reference to an entry in the data file (see Section 2.5), or to an expression defined in the Section [Expressions];
4. The linear-in-parameter utility function is composed of a list of terms, separated by a +. Each term is composed of the name of a parameter and the name of an attribute, separated by a ⋆. The parameter must be listed in Section [Beta], if it is a regular parameter. If it is a random parameter, the syntax is

   ```
   nameParam [ nameParam ]
   ```

   in the case of the normal distribution, or :

   ```
   nameParam { nameParam }
   ```

   to get a random parameter that comes from a uniform distribution. For example, in the case of the normal:

   ```
   BETA [ SIGMA ]
   ```

   Note that the blank after each name parameter is required. Also, parameters `BETA` and `SIGMA` have to be listed in Section [Beta]. In the context of an independent random parameter, `BETA` represents the mean while `SIGMA` corresponds to the standard deviation. With correlated random parameters, `SIGMA` technically corresponds to the appropriate term in the Cholesky decomposition matrix that captures the variance-covariance structure among the random parameters. For more details, see the technical section on Cholesky factorization An attribute must be an entry of the data file, or an expression defined in Section [Expressions]. In order to comply with this syntax, the Alternative Specific Constants must appear in a term like `ASC * one`, where `one` is defined in the Section [Expressions]. Here is an example:

   ```
   [Utilities]
   // Id Name  Avail  linear-in-parameter expression
     1   Alt1   av1   ASC1 * one + BETA1 [SIGMA] * x11 + BETA2 * x12
     2   Alt2   av2   ASC2 * one + BETA1 [SIGMA] * x21 + BETA2 * x22
   ```

```
3   Alt3   av3   ASC3 * one + BETA1 [SIGMA] * x31 + BETA2 * x32
4   Alt4   av4   ASC4 * one + BETA1 [SIGMA] * x41 + BETA2 * x42
5   Alt5   av5   ASC5 * one + BETA1 [SIGMA] * x51 + BETA2 * x52
6   Alt6   av6   ASC6 * one + BETA1 [SIGMA] * x61 + BETA2 * x62
```

If the utility function does not contain any part which is linear-in-parameters, then the keyword `$NONE` must be written. For example:

```
[Utilities]
// Id Name  Avail linear-in-parameter expression
  1   Alt1   av1  $NONE
```

[GeneralizedUtilities] This section enables the user to add nonlinear terms to the utility function. For each alternative, the syntax is simply the identifier of the alternative, followed by the expression. For example, if the utility of alternative 1 is

$$\beta_1 x_{11} + \beta_2 \frac{x_{12}^{\lambda} - 1}{\lambda},$$

the syntax is

```
[Utilities]
1 Alt1 av1 BETA_1 * X11

[GeneralizedUtilities]
1 BETA_2 * (X21 ^ LAMBDA - 1) / LAMBDA
```

Another example where a non-linear part is required is when specifying a log-normal random coefficient. Consult: Example LogNormal.

[ParameterCovariances] Biogeme 2.4 allows normally distributed random parameters to be correlated, and can estimate their covariance. By default, the variance-covariance matrix of the random parameters is supposed to be diagonal, and no covariance is estimated. If some covariances must be estimated, each pair of correlated random coefficients must be identified in this section. Each entry of the section should contain:

1. The name of the first random parameter in the given pair. If it appears in the utility function as `BETA [ SIGMA ]`, its name must be typed `BETA_SIGMA`.

2. The name of the second random parameter involved in the pair, using the same naming convention.
3. The default value that will be used as a starting point for the estimation;
4. The lower bound on the valid values;
5. The upper bound on the valid values;
6. The status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given value.

See Section 2.21 for an example. If no covariance is to be estimated, you must either entirely remove the section, or specify `$NONE` as follows:

```
[ParameterCovariances]
$NONE
```

[Draws] Number of draws to be used in Maximum Simulated Likelihood estimation.

[Expressions] In this section are defined all expressions appearing either in the availability conditions or in the utility functions of the alternatives defined in Section [Utilities]. If the expression is readily available from the data file, it can be omitted in the list. We show their use with the help of an example in Section 2.6. As we will discover later in the tutorial, it is good practice to generate new variables from this section especially when one objective is to compute market shares or to evaluate effects of policies with the help of Biosim 2.4.

We now summarize the syntax that can be used for generating new variables. Variables which form an expression might be of type float or of type integer. You can use numerical values or the name of a numerical variable. New variables can be created using unary and binary expression operators.

Unary expressions:

1. `y = sqrt(x)` `// y is square root of x.`
2. `y = log(x)` `// y is natural log of x.`
3. `y = exp(x)` `// y is exponential of x.`
4. `y = abs(x)` `// y is absolute value of x.`

binary expression: (Numerical)

1. `y = x + z // y is sum of variables x and z`
2. `y = x - z // y is difference of variables x and z`
3. `y = x * z // y is product of variables x by z`
4. `y = x / z // y is division of variable x by z`
5. `y = x ^ z // y is x to power of z (square would be y = x ^`
6. `y = x % z // y is x modulo z, i.e. rest of x/z`

binary expression: (Logical)

1. `y = x == z  // y is 1 if x equals  z, 0 otherwise`
2. `y = x != z  // y is 1 if x not equal to z, 0 otherwise`
3. `y = x || z  // y is 1 if x != 0 OR  z != 0, 0 otherwise`
4. `y = x && z  // y is 1 if x != 0 AND z != 0, 0 otherwise`
5. `y = x < z  // y is 1 if x < z  (note: also > )`
6. `y = x <= z  // y is 1 if x <= z  (note: also >= )`
7. `y = max(x,z)  // y is max of x and z  (note: also min)`

Note that an expression is considered to be TRUE if it is non zero, and FALSE if it is zero. For a full description of these expressions and alternative syntaxes, please look at the files `patSpecParser.y` and `patSpecScanner.l` in the BIOGEME distribution.

Loops can be defined if several expressions have almost the same syntax. The idea is to replace all occurrences of a string, say `xx`, by numbers. The numbers are generated within a loop, defined by 3 numbers: the start of the loop (`a`), the end of the loop (`b`) and the step (`c`) with the following syntax:

```
$LOOP {xx a b c}
```

The expression

```
$LOOP {xx 1 5 2} my_expression_xx = other_expression_xx * term_xx_fi
```

is equivalent to

```
my_expression_1 = other_expression_1 * term_1_first
my_expression_3 = other_expression_3 * term_3_first
my_expression_5 = other_expression_5 * term_5_first
```

Warning: make sure that the string is awkward enough so that it cannot match any other instance by mistake. For example, the loop

```
{xp 1 5 2} my_expression_xp = other_expression_xp * term_xp_first
```

is equivalent to

```
my_e1ression_1 = other_e1ression_1 * term_1_first
my_e3ression_3 = other_e3ression_3 * term_3_first
my_e5ression_5 = other_e5ression_5 * term_5_first
```

which is probably not the desired effect.

[Group] Provide here the formula to compute the group ID of the observed individual. Typically, a "`group`" entry will be available directly from the data file, but any formula can be used to compute it. Any expression described in Section [Expressions] is valid here. A different scale parameter will be estimated for the utility of each group.

[Exclude] Define an expression (see Section [Expressions]) which identifies entries of the data file to be excluded. If the result of the expression is not zero, the entry will be discarded.

[Model] Specifies which MEV model is to be used. Valid entries are `$BP` for Binary Probit, `$MNL` for Multinomial Logit model, `$NL` for single level Nested Logit model, `$CNL` for Cross-Nested Logit model and `$NGEV` for Network GEV model. See Section 1.1 for more details.

[PanelData] Used to specify the name of the variable (ex: `userID`) in the dataset identifying the observations belonging to a given individual and to specify the name of the random parameters that are invariant within the observation of a given individual `userID`. See the example at Section 2.20.

[Scale] A scale parameter is associated with each group. The utility function of each member of a group is multiplied by the associated scale parameter. A typical application is the joined estimation of revealed and stated preferences. It is therefore possible to estimate a MNL combining both data sources, without playing around with dummy nested structures as proposed by **?**. Each row of this section corresponds to a group. Five entries are required per row:

1. Group number: the numbering must be consistent with the group definition;
2. Default value that will be used as a starting point for the estimation (1.0 is a good guess);
3. Lower bound on the valid values;
4. Upper bound on the valid values;
5. Status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given value.

Clearly, one of the groups must have a fixed scale parameter.

[SelectionBias] Identifies the parameters capturing the selection bias, using the estimator proposed by Bierlaire et al. (2008). Each of them has to be listed in Section [Beta]. The section must contain a row per alternative for which a selection bias has to be estimated. Each row contains the number of the alternative and the name of the associated parameter. Note that these parameters play a similar role as the alternative specific constants, and must not be used with MNL.

```
[SelectionBias]
1 SB_1
4 SB_4
6 SB_6
```

[NLNests] This section is relevant only if the `$NL` option has been selected in Section [Model]. If the model to estimate is not a Nested Logit model, the section will be simply ignored. Note that multilevel Nested Logit models must be modeled as Network MEV models. Each row of this section corresponds to a nest. Six entries are required per row:

1. Nest name: the first character must be a letter (any case) or an underscore (_), followed by a sequence of letters, digits, underscore (_) or dashes (−), and terminated by a white space;

2. Default value of the nest parameter μ_m that will be used as a starting point for the estimation (1.0 is a good guess);
3. Lower bound on the valid values. It is usually 1.0, if μ is constrained to be 1.0. Do not forget that, for each nest i, the condition $\mu_i \geq \mu$ must be verified to be consistent with discrete choice theory;
4. Upper bound on the valid values;
5. Status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given value.
6. The list of alternatives belonging to the nest, numbered as specified in Section [Utilities]. Make sure that each alternative belongs to exactly one nest, as no automatic verification is implemented in Biogeme 2.4.

[CNLNests] This section is relevant only if the `$CNL` option has been selected in Section [Model]. If the model to estimate is not a Cross-Nested Logit model, the section will be simply ignored. Note that multilevel Cross-Nested Logit models must be modeled as Network MEV models. Each row of this section corresponds to a nest. Five entries are required per row:

1. Nest name: the first character must be a letter (any case) or an underscore (_), followed by a sequence of letters, digits, underscore (_) or dashes (−), and terminated by a white space;
2. Default value of the nest parameter μ_m that will be used as a starting point for the estimation;
3. Lower bound on the valid values. It is usually 1.0, if μ is constrained to be 1.0. Do not forget that, for each nest i, the condition $\mu_i \geq \mu$ must be verified to be consistent with discrete choice theory;
4. Upper bound on the valid values;
5. Status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given value.

[CNLAlpha] This section is relevant only if the `$CNL` option has been selected in Section [Model]. If the model to estimate is not a Cross-Nested Logit model, the section will be simply ignored. Each row of this section corresponds to a combination of a nest and an alternative. Six entries are required per row:

1. Alternative name, as defined in Section [Utilities];
2. Nest name: the first character must be a letter (any case) or an underscore (_), followed by a sequence of letters, digits, underscore (_) or dashes (-), and terminated by a white space;
3. Default value of the parameter capturing the level at which an alternative belongs to a nest that will be used as a starting point for the estimation;
4. Lower bound on the valid values (usually 0.0);
5. Upper bound on the valid values (usually 1.0);
6. Status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given value.

[Ratios] It is sometimes useful to read the ratio of two estimated coefficients. The most typical case is the value-of-time, being the ratio of the time coefficient and the cost coefficient. This feature is only implemented for fixed parameters. Computation of ratio of random parameters is not permitted in this version. Note that it is not straightforward to characterize the distribution of the ratio of two random coefficients. Ben-Akiva et al. (1993) suggest a simple approach that is directly implementable in BIOGEME to handle ratio of random parameters. Each row in this section enables to specify such ratios to be produced in the output file. Three entries are required:

1. The parameter (from Section [Beta]) being the numerator of the ratio;
2. The parameter (from Section [Beta]) being the denominator of the ratio;
3. The name of the ratio, to appear in the output file: the first character must be a letter (any case) or an underscore (_), followed by a sequence of letters, digits, underscore (_) or dashes (-), and terminated by a white space.

[ConstraintNestCoef] Since Version 0.2, it is possible to constrain nests parameters to be equal. This is achieved by adding to this section expressions like

```
NEST_A = NEST_B
```

where NEST_A and NEST_B are names of nests defined in Section [NLNests], Section [CNLNests] or Section [NetworkGEVNodes]. This section will become obsolete in future releases, as there is now a section for linear constraints on the parameters: (Section [LinearConstraints]).

[NetworkGEVNodes] This section is relevant only if the $NGEV option has been selected in Section [Model]. If the model to estimate is not a Network GEV model, the section will be simply ignored. Each row of this section corresponds to a node of the Network GEV model. All nodes of the Network GEV model except the root and the alternatives must be listed here, with their associated parameter. Five entries are required per row:

1. Node name: the first character must be a letter (any case) or an underscore (_), followed by a sequence of letters, digits, underscore (_) or dashes (−), and terminated by a white space;
2. Default value of the node parameter μ_j that will be used as a starting point for the estimation;
3. Lower bound on the valid values. It is usually 1.0. Check the condition on the parameters for the model to be consistent with the theory in Bierlaire (2002);
4. Upper bound on the valid values;
5. Status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given value.

[NetworkGEVLinks] This section is relevant only if the $NGEV option has been selected in Section [Model]. If the model to estimate is not a Network GEV model, the section will be simply ignored. Each row of this section corresponds to a link of the Network GEV model, starting from the a-node to the b-node. The root node is denoted by __ROOT. All other nodes must be either an alternative or a node listed in the section [NetworkGEVNodes]. Note that an alternative cannot be the a-node of any link, and the root node cannot be the b-node of any link. Six entries are required per row:

1. Name of the a-node: it must be either __ROOT or a node listed in the section [NetworkGEVNodes].
2. Name of the b-node: it must be either a node listed in the section [NetworkGEVNodes], or the name of an alternative.

3. Default value of the link parameter that will be used as a starting point for the estimation;
4. Lower bound on the valid values.
5. Upper bound on the valid values;
6. Status, which is 0 if the parameter must be estimated, or 1 if the parameter has to be maintained at the given value.

[LinearConstraints] In this section, the user can define a list of linear constraints, in one of the following syntaxes:

1. Formula = number,
2. Formula $\leq$ number,
3. Formula $\geq$ number.

The syntax is formally defined as follows:

```
oneConstraint : equation <= numberParam |
            equation = numberParam |
            equation >= numberParam
equation: eqTerm |
        - eqTerm |
        equation + eqTerm  |
        equation - eqTerm
eqTerm: parameter | numberParam * parameter
```

For example, the constraint

$$\sum_i \text{ASC}_i = 0.0$$

is written

```
ASC1 + ASC2 + ASC3 + ASC4 + ASC5 + ASC6 = 0.0
```

and the constraint

$$\mu \leq \mu_j$$

is written

```
MU - MUJ <= 0.0
```

or

```
MUJ - MU >= 0.0
```

[NonLinearEqualityConstraints] In this section, the user can define a list of nonlinear equality constraints of the form

$$h(x) = 0.0.$$

The section must contain a list of functions $h(x)$. For example, the constraint

$$\alpha_{a1}^{\mu_a} + \alpha_{b1}^{\mu_b} = 1$$

is written

```
[NonLinearEqualityConstraints]
ALPHA_A1 ^ MU_A  + ALPHA_B1 ^ MU_B - 1.0
```

[NonLinearInequalityConstraints] Biogeme 2.4 is not able to handle nonlinear inequality constraints yet. It should be available in a future version.

[DiscreteDistributions] Provide here the list of random parameters with a discrete distribution, or `$NONE` if there are none in the model. Each discrete parameter is described using the following syntax:

```
nameDiscreteParam < listOfDiscreteTerms >
```

where `nameDiscreteParam` is the name of the random parameter, and `listOfDiscreteTerms` is recursively defined as

```
oneDiscreteTerm |
listOfDiscreteTerms oneDiscreteTerm
```

where `oneDiscreteTerm` is defined as

```
nameValueParam ( nameProbaParam )
```

where `nameValueParam` is the name of the parameter capturing the discrete value of the random parameter, and `nameProbaParam` is the name of the parameter capturing the associated probability. Both must be defined in Section [Beta]. As an example,

```
[DiscreteDistributions]
BETA1 < B1 ( W1 ) B2 ( W2 ) >
```

defines a random parameter `BETA1`, which takes the value `B1` with probability (or weight) `W1`, and the value `B2` with probability `W2`. Note that for this to make sense, the constraint `W1 + W2 = 1.0` should be imposed (Section [LinearConstraints]). Note also that the parameter `BETA1` must not appear in Section [Beta].

[AggregateLast] Boolean which, for each row in the sample file, identifies if it is the last observation in an aggregate. Make sure that the value for the last row is nonzero. As all booleans in Biogeme 2.4, a numerical value of 0 means "FALSE" and a numerical value different from 0 means "TRUE". See section 2.15 for details. Any expression described in Section [Expressions] is valid here.

[AggregateWeight] Associates a weight to elemental observations of an aggregate. Corresponds to the term $P(\mathcal{C}_{\text{obs}}|i)$ in Eq. (2.12), Section 2.15. Any expression described in Section [Expressions] is valid here.

These sections are new in Biogeme 2.4:

[LaTeX] This section allows to define a description of each parameter to be used in the LaTeX file. For instance, the following section

```
[LaTeX]
ASC1   "Constant for alt. 1"
ASC2   "Constant for alt. 2"
ASC3   "Constant for alt. 3"
ASC4   "Constant for alt. 4"
ASC5   "Constant for alt. 5"
ASC6   "Constant for alt. 6"
BETA1  "$\beta_1$"
BETA2  "$\beta_2$"
```

will produce the following table:

Variable number	Description	Coeff. estimate	Robust Asympt. std. error	t-stat	p-value
1	Constant for alt. 2	-0.159	0.106	-1.49	0.13
2	Constant for alt. 3	-0.0869	0.111	-0.78	0.43
3	Constant for alt. 4	-0.511	0.172	-2.97	0.00
4	Constant for alt. 5	0.719	0.158	4.54	0.00
5	Constant for alt. 6	-1.39	0.195	-7.12	0.00
6	β_1	0.779	0.0301	25.85	0.00
7	β_2	0.810	0.0307	26.42	0.00

`[Derivatives]` This section is for advanced users only. Use it at your own risk.

When nonlinear utility functions are used, Biogeme 2.4 computes automatically the derivatives needed by the maximum likelihood procedure. However, this automatic derivation can significantly slow down the estimation process, as no simplification is performed. This section allows the user to provide Biogeme 2.4 with the analytical derivatives of the utility function, in order to speed up the estimation process. In some instances, half the estimation time was spared thanks to this feature.

A row must be provided for each combination of nonlinear utilities (defined in the Section Section [GeneralizedUtilities]) and parameters involved in the formula. Each of these rows contains three items:

- the identifier of the alternative,
- the name of the parameter,
- the formula of the derivative.

For instance, assume that the systematic utility of alternative 1 is

$$V_1 = \mathrm{ASC}_1 + \beta_1 \frac{(x_{11}+10)^{\lambda_{11}}-1}{\lambda_{11}} + \beta_2 \frac{(x_{12}+10)^{\lambda_{12}}-1}{\lambda_{12}}$$

so that

$$\frac{\partial V_1}{\beta_1} = \frac{(x_{11}+10)^{\lambda_{11}}-1}{\lambda_{11}}$$

$$\frac{\partial V_1}{\beta_2} = \frac{(x_{12}+10)^{\lambda_{12}}-1}{\lambda_{12}}$$

$$\frac{\partial V_1}{\lambda_{11}} = \beta_1\frac{(x11+10)^{\lambda_{11}}\lambda_{11}\ln(x_{11}+10)-(x_{11}+10)^{\lambda_{11}}+1}{\lambda_{11}^2}$$

$$\frac{\partial V_1}{\lambda_{12}} = \beta_2\frac{(x12+10)^{\lambda_{12}}\lambda_{12}\ln(x_{12}+10)-(x_{12}+10)^{\lambda_{12}}+1}{\lambda_{12}^2}$$

which is coded in Biogeme 2.4 as follows:

```
[Utilities]
// Id Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ..
  1   Alt1   av1   ASC1 * one
  .
  .
[GeneralizedUtilities]
1  BETA1 * ((x11 + 10 ) ^ LAMBDA11 - 1) / LAMBDA11 +
   BETA2 * ((x12 + 10 ) ^ LAMBDA12 - 1) / LAMBDA12

[Derivatives]
1 BETA1 ((x11 + 10 ) ^ LAMBDA11 - 1) / LAMBDA11
1 BETA2 ((x12 + 10 ) ^ LAMBDA12 - 1) / LAMBDA12
1 LAMBDA11
      BETA1 * ((x11 + 10) ^ LAMBDA11 * LN(x11 + 10) * LAMBDA11
             - (x11 + 10) ^ LAMBDA11 + 1) / (LAMBDA11 * LAMBDA11 )
1 LAMBDA12
      BETA2 * ((x12 + 10) ^ LAMBDA12 * LN(x12 + 10) * LAMBDA12
             - (x12 + 10) ^ LAMBDA12 + 1) / (LAMBDA12 * LAMBDA12 )
```

In addition to usual expressions, the formula may contain the following instruction:

```
$DERIV( formula , param )
```

which means that you ask Biogeme 2.4 to perform the derivation of the formula for you. Although it may be useful to simplify the coding of the derivatives, it is mandatory to use it for random parameters.

If `BETA [ SIGMA ]` is a random parameter, its derivative with respect to `BETA` is 1, but its derivative with respect to `SIGMA` cannot be written by the user, and must be coded

```
$DERIV( BETA [ SIGMA ] , SIGMA )
```

For instance, assume that the nonlinear utilities are defined as

```
1 exp( BETA1 [ SIGMA1 ] ) * x11
2 exp( BETA1 [ SIGMA1 ] ) * x21
```

The derivatives are coded as follows:

```
[Derivatives]
1 BETA1    exp( BETA1 [ SIGMA1 ] ) * x11
1 SIGMA1   exp( BETA1 [ SIGMA1 ] ) * x11
              * $DERIV( BETA1 [ SIGMA1 ] , SIGMA1 )
2 BETA1    exp( BETA1 [ SIGMA1 ] ) * x21
2 SIGMA1   exp( BETA1 [ SIGMA1 ] ) * x21
              * $DERIV( BETA1 [ SIGMA1 ] , SIGMA1 )
```

It is very easy to do an error in coding the analytical derivatives. If there is an error, Biogeme 2.4 will not be able to estimate the parameters, and will not even be able to detect that there is an error. Therefore, we strongly suggest to set the parameter `gevCheckDerivatives` to 1 and make sure that the numerical derivatives match sufficiently well the analytical derivatives. Also, estimate the model with few observations and few draws, once with and once without this section. The results should be exactly the same.

`[SNP]` This section allows to implement the test proposed by **?** (read the paper first if you are not familiar with the test). The section is composed of two things:

1. The name of the random parameter to be tested. If this parameter appears in the utility function as `BETA [ SIGMA ]`, its name in this section must be typed `BETA_SIGMA`.
2. A list of positive integers associated with a parameter. The integer is the degree of the Legendre polynomial, and the parameter the associated coefficient in the development. Note that the name of the parameter must appear in Section [Beta].

For instance, if parameter `BETA [ SIGMA ]` is tested using a seminon-parametric development defined by

$$1 + \delta_1 L_1(x) + \delta_3 L_3(x) + \delta_4 L_4(x),$$

the syntax in Biogeme 2.4 is

```
[Beta]
// Name  Value LowerBound UpperBound  status (0=variable, 1=fixed)
....
   BETA  0     -10000     10000       0
   SIGMA 1     -10000     10000       0
   SMP1  0     -10000     10000       0
   SMP3  0     -10000     10000       0
   SMP4  0     -10000     10000       0

[SNP]
// Define the coefficients of the series
// generated by the Legendre polynomials
BETA_SIGMA
1 SMP1
3 SMP3
4 SMP4
```

Note that only one random parameter can be transformed at a time.

`[OrdinalLogit]` The parameters of ordinal binary logit models (see Section 2.14) can be estimated. **However, this feature has not been fully tested, and should be seen as a prototype. Thank you for reporting any bug.** The segments of the utility difference space must be numbered in a sequential way, increasing from the leftmost to the rightmost. In this section, each segment must be associated with its lower bound, except the first (because its lower bound is $-\infty$). For instance, if there are 4 segments, like in Figure 2.3 on page 74, the following syntax is used:

```
[Beta]
....
tau1 0.3 -1000 1000 1
tau2 0.4 -1000 1000 0
tau3 0.5 -1000 1000 0

[OrdinalLogit]
```

```
1 $NONE    //  -infty --> tau1
2 tau1     //  tau1   --> tau2
3 tau2     //  tau2   --> tau3
4 tau3     //  tau3   --> +infty

[LinearConstraints]
tau1 - tau2 <= 0
tau2 - tau3 <= 0
```

Note that the constraints impose that the segments are well-defined. Recall also that the characters / / represent a comment in the file and they are not interpreted by Biogeme 2.4, as well as all remaining characters on the same line. Therefore, the following syntax for that section is completely equivalent:

```
[OrdinalLogit]
1 $NONE
2 tau1
3 tau2
4 tau3
```

However, we strongly advise to use comments in order to clearly identify the segments.

[SampleEnum] This section is ignored by BIOGEME. It is used by Biosim 2.4 and contains the number of simulations to perform in the sample enumeration step (see Section 2.13).

[ZhengFosgerau] This section is ignored by BIOGEME. It is used by Biosim 2.4 and contains instructions to perform the Zheng-Fosgerau specification test and residual analysis. Make sure to read the paper by **?** before using this section.

There is a line for each test, containing four items:

1. The first item defines the function `t` introduced by **?** to reduce the dimensionality of the test. It is typically either the probability of an alternative, or an expression involving coefficients and attributes of the models, as soon as the expression is continuous and not discrete. If it is a probability, the syntax is

   ```
   $P { AltName }
   ```

   where `AltName` is the name of the alternative as defined in Section [Utilities]. If it is a general expression, the syntax is

```
$E { expr }
```

where `expr` is an expression complying with the syntax of Section [Expressions]. However, it may also contain estimated parameters.

2. The second item is a parameter c used to define the bandwidth for the nonparametric regression performed by the test (see end of Section 2.1 in **?**). The bandwidth used by Biosim 2.4 is defined as $c/\sqrt{n}$, where n is the sample size. Most users will use the value $c = 1$.
3. The third and the fourth item are lower and upper bounds (resp.) Values of t outside of the bounds will not be used in the produced pictures. It is good practice to use wide bounds first, and to adjust them in order to obtain decent pictures. Note that if t is a probability, it does not make sense to have bounds wider and $[0 : 1]$.
4. The last item is the name of the function t, used in the report. Make sure to put the name between double-quotes.

Here is an example of the syntax:

```
[ZhengFosgerau]
$P { Alt1 } 1 0 1 "P1"
$E { x31 } 1 -1000 1000 "x31"
```

More details are available in Section 2.16.

[IIATest] This section is ignored by BIOGEME. It is used to compute the variables necessary to perform the McFadden omitted variables test on a subset of alternatives (see Eq. (2.13)).

The syntax is illustrated by the following example.

```
[IIATest]
// Description of the choice subsets to compute the new
// variable for McFadden's IIA test
// Name list_of_alt
C123 1 2 3
C345 3 4 5
```

Each row corresponds to a new variable. It consists in the name of the variable (it will appear as the column header in the output of Biosim 2.4), followed by the list of alternatives to be included in the associated subset.

2.5 Data file

Biogeme 2.4 assumes that each data file contains in its first line a list of labels corresponding to the available data, and that each subsequent line contains the exact same number of numerical data, each row corresponding to an observation. Delimiters can be tabs or spaces. Note that missing values must not be represented by dots. Instead, replace them by obviously meaningless values, defined by `gevMissingValue`. For those who have used it, the convention was the same in the HieLoW package (see **?**, **?**).

WARNING: if you have created a data file on DOS or Windows, it may cause problems. If you work in a Windows environment and want to avoid using Emacs, we recommend using `TextPad` which is very intuitive to Windows users. Then just make sure you save the file in a UNIX format by selecting the UNIX format in the Save As window. The users working under Linux must convert the file with a utility like `dos2unix`, available from

`www.megaloman.com/~hany/software/hd2u`,

or using Emacs. With GNU Emacs 20.7.1, a (DOS) tag appears at the left of the Emacs info bar when the file is edited, indicating that the file needs to be converted. Use the menu `Mule|Set Coding System|Buffer File` or type[2] `C-x RET f`. Emacs asks you to choose a

`"Coding system for visited file (default, nil)"`.

Choose the default by hitting the return key and save the file.

2.6 Data transformation

In this section, we use a basic example where the variables x11 to x61 and x12 to x62 are available on the dataset. Let us use the syntax available in the `Expression` section of the input file to create new variables to be used in the model. The following sections of an input file provide an examples of some new variables that can be created using the `Expression` section. If one wishes to use Biosim 2.4 to produce the predicted probabilities in order to perform post estimation analysis, it is a good idea to create new variables in the [Expressions] section. Assume that the idea is to see how choice probabilities would vary as

[2] In Emacs terminology, `C-x` means that you must press the `Ctrl` key and the `x` key together.

the value of x21 is increased, for instance. To get the right calculation, because x21 is involved in the new variables, the change would correctly disseminate.

```
[Beta]
// Name Value  LowerBound UpperBound  status (0=variable, 1=fixed)
ASC1        0         -10000         10000       1
ASC2        0         -10000         10000       0
ASC3        0         -10000         10000       0
ASC4        0         -10000         10000       0
ASC5        0         -10000         10000       0
ASC6        0         -10000         10000       0
BETA1       0         -10000         10000       0
BETA2       0         -10000         10000       0
GAMMA1      0         -10000         10000       0
GAMMA2      0         -10000         10000       0
[Utilities]
// Id Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ...
  1   Alt1   av1   ASC1 * one + BETA1 * x11 + BETA2 * x12
                              + GAMMA1 * x11sq   + GAMMA2 * dum12
  2   Alt2   av2   ASC2 * one + BETA1 * x21 + BETA2 * x22
                              + GAMMA1 * x21sq   + GAMMA2 * dum12
  3   Alt3   av3   ASC3 * one + BETA1 * x31 + BETA2 * x32
                              + GAMMA1 * x31sq
  4   Alt4   av4   ASC4 * one + BETA1 * x41 + BETA2 * x42
                              + GAMMA1 * x41sq
  5   Alt5   av5   ASC5 * one + BETA1 * x51 + BETA2 * x52
                              + GAMMA1 * x51sq
  6   Alt6   av6   ASC6 * one + BETA1 * x61 + BETA2 * x62
                              + GAMMA1 * x61sq
[Expressions]
// Define here arithmetic expressions for name that are not directly
// available from the data
one = 1
         // Loop over alternatives 1 to 6 to create the square of xi1
{zzz 1 6 1}  xzzz1sq = xzzz1 ^ 2

         // Create dum12 = 1 if x11 >= 1 or x21 >= 1, 0 otherwise.
dum12 = ( x11 >= 1 ) || ( x21 >= 1 )
```

2.7 Statistics

The file containing the statistics of the sample is `mymodel.sta` It contains the following information.

1. The sample size and the sum of all weights are reported. If they don't match, Biogeme 2.4 suggests a factor to modify the weights:

   ```
   --> It is recommended to multiply all weights by 1.45678
   ```

   In that case, you may want to modify the weight definition in the model specification file:

   ```
   [Weight]
    weight *  1.45678
   ```

2. The number of excluded observations, due to the condition defined in Section [Exclude] of the model specification file, is reported.

3. The total number of observations in the file

4. The number of cases, which is the number of alternatives available to each observation minus the number of observations (see Ben-Akiva and Lerman, 1985, p. 90).

5. For each attribute, the mean, the minimum and the maximum value across the sample are reported.

6. The number of chosen alternatives, both not taking and taking the weight into account.

7. The group membership, both not taking and taking the weight into account.

2.8 Report file

The report file (`mymodel.rep`) contains the results of the maximum likelihood estimation of the model. First, general information is reported:

- Type of model which has been estimated.
- Sample size.
- `Null log-likelihood` is the log-likelihood of the sample for a Multinomial Logit model where all β parameters are 0. It is computed as

$$\mathcal{L}^0 = \sum_{n \in \text{sample}} \omega_n \ln \frac{1}{\#\mathcal{C}_n} \tag{2.4}$$

  where $\#\mathcal{C}_n$ is the number of alternatives available to individual n and ω_n is the associated weight.

- `Cte log-likelihood` is the log-likelihood of the sample for a Multinomial Logit model where the only coefficients are the alternative specific constants. **If all alternatives are always available**, it is computed as

$$\sum_{j \in \mathcal{C}} n_j \ln n_j - n \ln n, \tag{2.5}$$

  where n_j is the number of times alternative j has been chosen, and $n = \sum_{j \in \mathcal{C}} n_j$ is the number of observations in the sample. Note that if some alternatives are not available for some observations, the formula is not valid, and the value is not reported.

- `Init log-likelihood` is the log-likelihood of the sample for the model defined in the `.mod` file.
- `Final log-likelihood` is the log-likelihood of the sample for the estimated model.
- `Likelihood ratio test` is

$$-2(\mathcal{L}^0 - \mathcal{L}^*) \tag{2.6}$$

  where $\mathcal{L}^0$ is the log-likelihood of the sample for a Multinomial Logit model where all β parameters are 0, defined by (2.4), and $\mathcal{L}^*$ is the log-likelihood of the sample for the estimated model.

- `Rho-square` is
$$\rho^2 = 1 - \frac{\mathcal{L}^*}{\mathcal{L}^0}. \tag{2.7}$$

- `Adjusted rho-square` is
$$\rho^2 = 1 - \frac{\mathcal{L}^* - K}{\mathcal{L}^0}. \tag{2.8}$$
where K is the number of estimated parameters. Note that this statistic is meaningless in the presence of constraints, where the number of degrees of freedom is less than the number of parameters.

- `Final gradient norm` is the gradient of the log-likelihood function computed for the estimated parameters. If no constraint is active at the solution, it should be close to 0. If there are equality constraints, or if some bound constraints or inequality constraints are active at the solution (that is, they are verified with equality), the gradient may not be close to zero.

- `Diagnostic` is the diagnostic reported by the optimization algorithm. If the algorithm has not converged, the estimation results presented in the file cannot be used as such.

- `Iterations` is the number of iterations used by the algorithm before it stopped.

- `Run time` is the actual time used by the algorithm before it stopped.

- `Variance-covariance` specifies how the second-derivative matrix (inverted to obtain the variance-covariance matrix) has been calculated. If can be either from a finite difference approximation (which is accurate, but may take time to compute), or from the BHHH matrix (which is less accurate, but faster to compute, see Berndt et al., 1974). The user selects this option with parameter `gevVarCovarFromBHHH`.

Then follow results about the parameters.

- The estimated value of the β parameters, with the associated standard error, the t-test and the corresponding p-value. A sign (defined by `gevWarningSign`) is appended if the t-test fails, according to the threshold specified by the parameter `gevTtestThreshold` in the parameter file. Similar values

obtained from the robust estimation of the variance-covariance matrix are also provided (see Section 1.2).

- The estimated value of the μ parameter, with the associated standard error and the t-test. A sign (defined by `gevWarningSign`) is appended if the t-test fails, according to the threshold specified by the parameter `gevTtestThreshold` in the parameter file. Similar values obtained from the robust estimation of the variance-covariance matrix are also provided (see Section 1.2).

- The estimated value of the GEV model parameters, with the associated standard error and the t-test. A sign (defined by `gevWarningSign`) is appended if the t-test fails, according to the threshold specified by the parameter `gevTtestThreshold` in the parameter file. Similar values obtained from the robust estimation of the variance-covariance matrix are also provided (see Section 1.2). Note that the t-test is computed to compare the estimated value both to 0 and 1.

- The estimated value of the scale parameters, with the associated standard error and the t-test. A sign (defined by `gevWarningSign`) is appended if the t-test fails, according to the threshold specified by the parameter `gevTtestThreshold` the parameter file. Similar values obtained from the robust estimation of the variance-covariance matrix are also provided (see Section 1.2). Note that the t-test is computed to compare the estimated value to 1.

- The ratios requested in Section [Ratios] of the model specification file.

- A covariance/correlation analysis of pairs of estimated β parameters, sorted according to the t-test value. A sign (defined by `gevWarningSign`) is appended if the t-test fails, according to the threshold specified by the parameter `gevTtestThreshold` in the parameter file.

- When applicable, a report about the singularity of the second derivatives matrix.

2.9 Summary file

This file is designed to be imported into a spreadsheet, for further analysis, results comparisons and presentations. Each line corresponds to one run of Biogeme 2.4 in the directory. They contain

1. The date and time when the run has terminated;
2. The name of the model;
3. The name of the report file;
4. The final log-likelihood;
5. The sample size;
6. The estimated value and t-test of the parameters listed in the file defined by `gevSummaryParameters`;
7. The exclusion condition.

2.10 Other output files

Biogeme 2.4 also generates files for debugging purposes.

1. `parameters.out` This file contains the list of parameters that Biogeme 2.4 has actually used during the run, and the associated values. Syntax errors in the file `default.par` are usually not detected by Biogeme 2.4. Instead, in the presence of a syntax error, the rest of the file is skipped without warning and default values of the parameters are then used. This has to be improved in the future. At this point, the file `parameters.out` helps to check if the values actually used by Biogeme 2.4 are the values that were set by the user.
2. `model.debug` This file contains debugging information about the model specification. A regular user should not need it.
3. `mymodel.res [res]` This file has exactly the same structure as the file `mymodel.mod`, where the default values of the parameters have been replaced by the estimated values. This file is a valid model specification

file for a future Biogeme 2.4 run. It just needs to be renamed with a .mod extension.

When requested (see parameter `gevDumpDrawsOnFile`), the draws are dumped into a file. They are organized as follows: for each observation, for each random parameter, there are n draws generated, where n is defined in the Section [Draws]. Those draws are either pseudo-random numbers or Halton sequences, depending on the value of `gevRandomDistrib`).

2.11 Optimization algorithms

Biogeme 2.4 can use five different optimization algorithms: CFSQP, DONLP2, SOLVOPT, BIO and BIOMC. Note that it is possible that each of them produce different solutions. Usually, the discrepancies are small, and due to numerical differences and various stopping criteria. Also, none of them identifies a global maximum of the likelihood function. Therefore, it may happen that one of them is caught in a local maxima, different from local maxima found by other algorithms.

Which one to choose is difficult to say. They all have advantages and disadvantages. BIO (which stands for BIerlaire's Optimization, like in BIOGEME) has been specifically adapted for this software package, but it cannot accommodate nontrivial constraints yet. A version for simulated maximum likelihood estimation, called BIOMC, is also available. CFSQP is not free and, therefore, not included in the general distribution of Biogeme 2.4. DONLP2 is slower than CFSQP, but faster than SOLVOPT. SOLVOPT can sometimes be very slow. The algorithm is not to blame. It has not been originally designed for the kind of optimization problem involved in BIOGEME. However, it seems robust on ill-specified models. When other algorithms fail to converge, SOLVOPT may succeed in finding a solution. Therefore, our recommendation would be:

- If there are no non-trivial constraint on the parameters, use BIO.
- If there are, or if BIO is very slow, use CFSQP if you have it.
- If not, use DONLP2.
- If all fail, try SOLVOPT.
- If it fails again, redefine your model.

It is always a good idea to solve the same problem with several algorithms.

BIO BIO is a trust-region algorithm (see Conn et al., 2000) designed for problems with simple bounds constraints, using a truncated conjugate-gradient method to solve the trust-region subproblem. By default, it uses the BHHH (Berndt et al., 1974) matrix as an approximation of the second derivatives matrix. At each iteration, it displays

1. The value used in the stopping criterion at iterate x, that is the infinite norm of the relative gradient, computed as

$$\gamma = \max_i \left| \frac{|g_i(x_i)| \max(1, |x_i|)}{\max(f(x), t_f)} \right| \tag{2.9}$$

   where g is the projection of the gradient of f onto the feasible set, t_f is defined by `BTRTypf` (see Dennis and Schnabel, 1996).
2. The iteration number;
3. The radius of the trust region;
4. The current value of the objective function[3];
5. The exit status of the subproblem solver;
6. The value used to check if the radius of the trust region is appropriate, that is

$$\frac{f(x^+) - f(x_c)}{m(x^+) - m(x_c)}$$

   where x_c is the current iterate, x^+ the new candidate, and m the quadratic model approximating f at x_c (see Conn et al., 2000 for details).
7. The number of variables not constrained to one of their bounds:
8. The status of the iteration, that is very successful (++), successful (+) or unsuccessful (-).
9. If the subproblem is preconditioned, a `P` is also displayed.

BIOMC is a version of BIO designed for simulated maximum likelihood. The idea is to use BIO with few draws in the beginning, in order to obtain a

[3] As it is a minimization algorithm, the objective function is the opposite of the log-likelihood and, therefore, is a positive value.

rough value of the parameters, and then to increase the number of draws until reaching the number required by the user (d_u). Let d_k be the number of draws considered by BIOMC (where d_0 is defined by `BTRStartDraws`). Then, algorithm BIO runs until the following condition is verified

$$\frac{\ln \gamma}{\ln \varepsilon} \geq \frac{d_k}{d_u},$$

where γ is defined by (2.9) and ε is defined by `BTRTolerance`, so that the convergence requirements are not as strong when the number of draws is low. Then, the number of draws is increased

$$d_{k+1} = \min(\lambda d_k, d_u),$$

where the factor λ is defined by `BTRIncreaseDraws`, and the process starts again. The algorithm stops when $d_k = d_u$.

In general, BIOMC is not faster than BIO. But it allows to obtain good approximations of the parameters pretty quickly.

CFSQP CFSQP is a C implementation of the FSQP optimization algorithm developed by E.R. Panier, A.L. Tits, J.L. Zhou, and C.T. Lawrence (see **?**). CFSQP is licensed to AEM Design. The conditions for external use are the following

1. The CFSQP routines may not be distributed to third parties. Interested parties shall contact AEM Design directly.
2. If modifications are performed on the routines, these modifications shall be communicated to AEM Design. The modified routines will remain the sole property of the authors.
3. Due acknowledgment shall be made of the use of the CFSQP routines in research reports or publications. Whenever such reports are released for public access, a copy shall be forwarded to AEM Design.
4. The CFSQP routines may only be used for research and development, unless it has been agreed otherwise with AEM Design in writing.

If you have a CFSQP license, and need a Windows version of Biogeme 2.4 with CFSQP, send an Email to (`michel.bierlaire@epfl.ch`) to receive the executable.

SOLVOPT Solvopt is defined by its authors, Alexei V. Kuntsevich and Franz Kappel, as follows: *The program SolvOpt (Solver for local optimization problems) is concerned with minimization resp. maximization of nonlinear, possibly non-smooth objective functions and with the solution of nonlinear programming problems taking into account constraints by the so-called method of exact penalization.* The package implements a version of minimization method with space dilation by **?**. See **?** for a tutorial of the package and the description of the algorithm.

DONLP2 DONLP2 is a sequential equality constrained quadratic programming method, developed by **?**. The algorithm is described by **?** and **?**. The conditions of use of the DONLP2 package are the following.

1. donlp2 is under the exclusive copyright of P. Spellucci

   e-mail:spellucci@mathematik.tu-darmstadt.de

   "donlp2" is a reserved name
2. donlp2 and its constituent parts come with no warranty, whether expressed or implied, that it is free of errors or suitable for any specific purpose. It must not be used to solve any problem, whose incorrect solution could result in injury to a person , institution or property. It is at the users own risk to use donlp2 or parts of it and the author disclaims all liability for such use.
3. donlp2 is distributed "as is". In particular, no maintenance, support or trouble-shooting or subsequent upgrade is implied.
4. The use of donlp2 must be acknowledged, in any publication which contains results obtained with it or parts of it. Citation of the authors name and netlib-source is suitable.
5. The free use of donlp2 and parts of it is restricted for research purposes commercial uses require permission and licensing from P. Spellucci.

2.12 Generating draws

All distributions needed by Biogeme 2.4 are generated from uniform [0,1] distributions. The derivation of uniform [-1,1] is obvious. The derivation of normal $N(0,1)$ is performed using Wichura's method **?**.

The uniform [0,1] distributions can be generated in three different ways. See Train, 2009 for more details.

1. Using Unix's pseudo-random number generator.
2. Using Halton draws.
3. Using the Modified Latin Hypercube Sample procedure proposed by Hess et al. (2006). It generates a small random perturbation of equally distributed draws. If R is the number of draws required, it generates a vector $d(0:R-1)$ such that
$$d(i) = \frac{i}{R} + \frac{\xi}{R}$$
where ξ is a draw from a uniform [0:1] distribution, such that $\xi \neq 0$ and $\xi \neq 1$.

2.13 Simulation: sample enumeration

The package Biosim 2.4 is invoked exactly like Biogeme 2.4, with the exact same input file. But instead of performing a parameter estimation, it performs a sample enumeration. Sample enumeration performed by Biosim 2.4, produces correct predicted probabilities for all model versions as long as it is not in a panel data setting. The panel data setting requires a large set of choice probability calculations and this will not be available in Biosim 2.4 in the very near future.

The file `mymodel.enu` contains the result of the sample enumeration. For each observation in the sample, the following results are provided:

1. The identifier of the choice actually reported in the sample file;
2. The name of the choice actually reported in the sample file;
3. The probability given by the model for the chosen alternative;
4. For each alternative, the utility given by the model;
5. For each alternative, the probability given by the model;
6. A list of simulated choice, based on Monte-Carlo simulation using the model.

The sample enumeration file is extremely useful for producing the aggregate market shares computed at convergence. The usual way to produce the predicted probabilities is to rename the file that is generated at convergence with a .res extension into a file with a .mod extension. Then replace the word biogeme with biosim in the call. For a model input file called `mymodel.mod` for which we rename the `mymodel.res` file into `mymodel_res.mod` and given a data base called `sample.dat`, the command:

```
biosim mymodel_res  sample.dat
```

`mymodel_res.enu` which would contain among other things the choice probabilities of each available alternatives. To produce an analysis of the impact of a given policy, change the file `mymodel_res.enu` to reflect the change to a given variable or a given set of variable. This is where using the [Expressions] section to create variables takes all its sense. Applying Biosim 2.4 to this input file would create a `.enu` file that would reflect the new values of the choice probabilities following the change. Then, bringing the two `.enu` into an Excel spreadsheet, for example, would allow one to measure the change of aggregate shares computed by given market segments.

As a final note, the use of weights may be counter-intuitive in Biosim. If an observation is weighted by ω_n, its contribution to the log-likelihood in the estimation process would be

$$\omega_n \ln P_n(i_n) = \ln P_n(i_n)^{\omega_n}.$$

As Biosim and Biogeme use the exact same formulation, the use of weights with Biosim produces the value

$$P_n(i_n)^{\omega_n},$$

which may not be the desired effect.

2.14 Ordinal logit

An ordinal binary choice model is derived when ordinal responses are available, where the respondent not only reports the preference, but also the strength of the preference. For instance, if alternatives i and j are available, the respondent can report one of the following.

- definitely choose j;
- probably choose j;

- indifferent;
- probably choose i;
- definitely choose i.

As for the binary choice model, the selected category is explained by the difference $U_{in} - U_{jn}$ between the utilities of the two alternatives, as depicted in Figure 2.3.

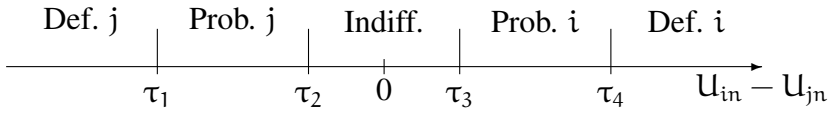

Figure 2.3: Categories for the ordinal binary choice model

Formally, we consider $Q \geq 2$ categories, ordered such that category q corresponds to a stronger preference towards alternative i compared to category $q-1$, for $q = 1, \ldots, Q$. We define $Q+1$ parameters τ_q, $q = 0, \ldots, Q$, such that $\tau_0 = -\infty$, $\tau_Q = +\infty$, and $\tau_{q-1} \leq \tau_q$, $q = 1, \ldots, Q$. A category q is associated with the interval $[\tau_{q-1}, \tau_q]$. The probability for category q to be selected by the respondent is

$$\begin{aligned} P_n(q) &= \Pr(\tau_{q-1} \leq U_{in} - U_{jn} \leq \tau_q) \\ &= \Pr(\tau_{q-1} \leq (V_{in} - V_{jn}) - (\varepsilon_{jn} - \varepsilon_{in}) \leq \tau_q) \\ &= \Pr(V_{in} - V_{jn} - \tau_q \leq \varepsilon_n \leq V_{in} - V_{jn} - \tau_{q-1}) \\ &= F_{\varepsilon_n}(V_{in} - V_{jn} - \tau_{q-1}) - F_{\varepsilon_n}(V_{in} - V_{jn} - \tau_q) \end{aligned} \tag{2.10}$$

where $\varepsilon_n = \varepsilon_{jn} - \varepsilon_{in}$, and F_{ε_n} is the CDF of ε_n. By definition of the CDF, the probability for the extreme categories simplify to

$$\begin{aligned} P_n(1) &= F_{\varepsilon_n}(V_{in} - V_{jn} + \infty) - F_{\varepsilon_n}(V_{in} - V_{jn} - \tau_1) \\ &= 1 - F_{\varepsilon_n}(V_{in} - V_{jn} - \tau_1), \\ P_n(Q) &= F_{\varepsilon_n}(V_{in} - V_{jn} - \tau_{Q-1}) - F_{\varepsilon_n}(V_{in} - V_{jn} - \infty) \\ &= F_{\varepsilon_n}(V_{in} - V_{jn} - \tau_{Q-1}). \end{aligned} \tag{2.11}$$

In particular, if ε_n is logistically distributed, we obtain the *ordinal logit* model. We immediately note that binary choice models are specific instances of ordinal binary choice models, with two categories ($Q = 2$), and $\tau_1 = 0$.

The parameter τ can be estimated by Biogeme 2.4 using Section [Ordinal-Logit]

Example

The file `23ordinalLogit.mod` contains an example of such a model, using 4 intervals (and, therefore, three thresholds τ_1, τ_2 and τ_3). In order to comply with the general syntax of Biogeme 2.4, the following conventions are used:

- the model must contain exactly two alternatives;
- the identifier of the alternative is completely ignored;
- alternative `i` is the first by alphabetical order, and alternative `j` is the second.

To illustrate this last point, the example is designed with a xonfusing numbering. "`Alt2`" is first if we rank the alternatives using the IDs, and second if we rank them by alphabetical order. As the IDs are ignored, the model is based on `Alt1 - Alt2`, so that

- V_{in} = `ASC1 * one + BETA1 * x11 + BETA2 * x12`,
- V_{jn} = `ASC2 * one + BETA1 * x21 + BETA2 * x22`.

In order to remove any ambiguity, the exact formula used by the model (that is, the difference of the two utilities) is reported in the output file.

Note that if all the threshold parameters are estimated, all constants must be normalized to 0 (see the example).

2.15 Latent choice

A choice is said to be "latent" when it is not directly observed. This idea has been proposed by **?** in a route choice context where the actual chosen route was not directly observed. Instead, the respondent reported a sequence of locations that they traversed. In many cases, several routes in the network may have produced the same reported locations.

Each observation consists of an aggregate, a set of actual alternatives that may correspond to the observed situations. If $\mathcal{C}_{\text{obs}}$ is the observed aggregate, than the probability given by the choice model is

$$\mathrm{P}(\mathcal{C}_{\text{obs}}) = \sum_{\mathrm{i}\in\mathcal{C}} \mathrm{P}(\mathcal{C}_{\text{obs}}|\mathrm{i})\mathrm{P}(\mathrm{i}|\mathcal{C}). \tag{2.12}$$

Figure 2.4: Density of Util2 = ((ASC2 * one) + (BETA1 * x21)) + (BETA2 * x22)

Equation $P(\mathcal{C}_{\text{obs}}|i)$ can be viewed as a measurement equation, and represents the probability to observe $\mathcal{C}_{\text{obs}}$ if i was the actual choice.

In Biogeme 2.4, an aggregate observation is represented by a consecutive sequence of elemental observations, associated with the probability $P(\mathcal{C}_{\text{obs}}|i)$. Two additional sections in the model specification file are used for the specification: section [AggregateLast] defines a boolean with is true if the corresponding row is the last elemental observation of the current aggregate, and false otherwise. section [AggregateWeight] defines the value of $P(\mathcal{C}_{\text{obs}}|i)$.

2.16 The Zheng Fosgerau test

Biosim 2.4 can compute the Zheng-Fosgerau test. Proposed by **?**, it has been adapted to discrete choice models by **?**. In addition to the value of the test itself, Biosim 2.4 reports pictures allowing to perform residual analysis.

We consider here the example in the file `24ZhengFosgerau.mod`. Note that this is not a genuine model. Our objective here is to illustrate the tool. In this file, 3 tests are proposed, with 3 definitions for the function t introduced by **?**. We consider the second one, which is exactly the expression of the utility function of alternative 2. The syntax is

```
$E { ASC2 * one + BETA1 * x21 + BETA2 * x22 } 1 -1000 1000 "Util2"
```

Biosim 2.4 first generates a plot to show how t (i.e. the utility of alternative 2 in this case) is distributed in the sample. This plots appears in the file `24ZhengFosgerau_zheng.tex`, which must be processed with the LaTeX[4] word processor. The density function is estimated using nonparametric regression. As we can see on Figure 2.4, the shape is not too different from a normal distribution, with very few values of this expression are out of the range $[-10 : 10]$.

Biosim 2.4 also generates plots of the residuals, that is the difference $(y_{in} - P_n(i))$, where y_{in} is 1 if individual n has chosen alternative i in the sample, and

[4] LaTeX is distributed freely on internet. Note that the package `pstricks` is required to produce the plots.

Figure 2.5: Testing Util2 = ((ASC2 * one) + (BETA1 * x21)) + (BETA2 * x22) with alt. Alt2

Figure 2.6: Testing Util2 in $[-7 : 7]$ with alt. Alt2

0 otherwise, and $P_n(i)$ is the probability as computed by the model. In principle, the residuals should be a white noise unrelated to the independent variables. Figure 2.5 plots a nonparametric estimation of the residuals for alternative 2 as a function of t, that is the utility of alternative 2 in this case. Clearly, the residuals are not independent from the utility, which is a sign of a misspecification. Unfortunately, the source of the misspecification itself is not revealed by the plot.

We can trim the data and exclude values below -7 and above 7, using the syntax

```
$E { ASC2 * one + BETA1 * x21 + BETA2 * x22 } 1 -7 7 "Util2"
```

Doing this, we exclude only 2.8% of the data, and the plot on Figure 2.6 emphasizes even more the misspecification. Note that trimming is not equivalent to a zoom on the plot. The data are excluded before the nonparametric estimation is performed. Although in principle it should not affect the general shape of the plot, some discrepancies my appear especially at the borders of the selected interval.

If LaTeX is not available, the plots may be generated using Excel or something similar. Indeed, Biosim 2.4 creates also the file `24ZhengFosgerau_zheng.enu` which is an ASCII file, easily imported in a spreadsheet. The file is organized as follows. For each test (that is for each function t defined by the user), a total of $5(J + 1)$ rows are generated, where J is the number of alternatives. The name of the function t is reported in the first column. The first 5 rows contain the smallest and the largest values, the range, the bandwidth used for the nonparametric regression (that is $c/\sqrt{n}$, where n is the sample size and c is user defined, and 1 by default), and a description of the trimming using the syntax

```
[l:u]: nl < ni > nu <==> pl% < pi% > pu%
```

where l and u are the lower and the upper bound defined by the user, `nl` and `nu` are the number pieces of data excluded because they are beyond the lower

(resp. upper) bound, and `ni` is the number of pieces of data considered in the analysis. `pl, pi` and `pu` report the same information in terms of percentages.

For each alternative, 5 rows are generated, containing the following information:

1. The name of the alternative, the ID and the value of the Zheng test;
2. A list of values corresponding to th x-axis of the plot. By default, there are 100 of them.
3. A list of values for the residual, corresponding to the continuous line in Figure 2.6.
4. A list of values for the lower bound of the confidence interval, corresponding to the lower dashed line in Figure 2.6.
5. A list of values for the upper bound of the confidence interval, corresponding to the upper dashed line in Figure 2.6.

2.17 IIA test

Suppose that we have estimated a logit model, using all the observations. The final log likelihood of this model is $\mathcal{L}_1$. Denote by P_{in} the probability given by this model that individual n in the sample chooses alternative i.

Consider $\widehat{\mathcal{C}} \subseteq \mathcal{C}$ a given subset of alternatives. Define the new variables

$$z_{in} = \begin{cases} V_{in} - \dfrac{\sum_{j \in \widehat{\mathcal{C}}} P_{jn} V_{jn}}{\sum_{j \in \widehat{\mathcal{C}}} P_{jn}} & \text{if } i \in \widehat{\mathcal{C}}, \\ 0 & \text{if } i \notin \widehat{\mathcal{C}}. \end{cases} \tag{2.13}$$

Estimate the same model as before where the new variables have been also included in the specification. Testing if IIA holds is equivalent to testing if all the coefficients of the new variables are 0, which can be performed with a likelihood ratio test.

2.18 Merging files

It is often convenient (in particular when performing the IIA test described in Section 2.17) to expand the original observation database with new variables that have been computed (for example) with Biosim 2.4. A simple utility, called `biomerge` has been implemented to perform this task easily. If `file1`, `file2`, ..., `filen` are n ASCII files containing the exact same number of rows each. The command

```
biomerge file1 file2 ... filen
```

will generate a file called `biomergeOutput.lis` such that row number j of this file is the concatenation of row number j of all input files.

2.19 Known problems

1. The loglikelihood estimator used by Biogeme 2.4 to estimate a model with panel data has been designed for instances where all random parameters are panel. When some of the random parameters are cross-sectional, the estimator is not exactly correct. It is conjectured that the estimates are still consistent, but not as efficient as they should be. So, except if you have good reasons to do otherwise, make sure that all the random variables are listed in the `[PanelData]` section.

2. The values in the section "Variance of random coefficients" have been reported wrong by several users on some model instances. Make sure to verify if the reported values make sense. If not, simply ignore this section. The value of the parameters reported in the "Utility parameters" section are correct.

2.20 Basic examples

A list of examples is available from the BIOGEME webpage. Two complementary data files are available: `sample.dat` containing 1000 observations, and `sample2.dat` containing 999 observations. These two files are grouped in the file `fullsample.dat`. Therefore, the following runs are equivalent:

```
biogeme mymodel sample.dat sample2.dat
```

and

```
biogeme mymodel fullsample.dat
```

These examples are designed to help the user understanding how to use Biogeme 2.4. They are not meant to illustrate how to build good models. Note that reading this section supposes that you can read and edit the example files, and run Biogeme 2.4.

A binary logit model

`00bl.mod`

Example of a model specification file for binary logit. The Alternative Specific Constant (ASC) associated with alternative 1 has been fixed, and will not be estimated.

In order to use the same data file as for the other examples, all observations with a choice strictly larger than 2 are excluded using the following syntax:

```
[Exclude]
Choice > 2
```

Therefore, each observation corresponds to the choice of alternative 1 or alternative 2.

A binary probit model

`00bp.mod`

Example of a model specification file for binary probit. Except for the distribution assumption on the error term, the model specification is the same as in `00bl.mod`.

A basic model

`01mnl-basic.mod`

The file `01mnl-basic.mod` contains the minimum model description needed by Biogeme 2.4. It is a typical example of a file created by hand. The model has 6 alternatives, and the utility functions contain only the Alternative Specific Constants. The ASC associated with alternative 1 has been fixed, and will not be estimated.

Weight the observations

02mnl-weights.mod

The observations are now weighted. But weighting the observations must be done carefully. Namely, the sum of all weights should be equal to the sample size. Biogeme 2.4 provides some help to achieve this property. In the file 02mnl-weights.sta, it is reported

```
Sample size=1000
Total weight=994.295
     --> It is recommended to multiply all weights by 1.005738e+00
```

Corrected weights

03mnl-weights.mod

The weights are corrected based on the recommendation from the .sta file in the following way:

```
[Weight]
Weight * 1.005738
```

Heterogeneous samples

04mnl-heterosample.mod

This example illustrates the estimation of scale parameters for different groups in the sample. We have here two groups, defined by

```
[Group]
(Id <= 50) * 1 + (Id >= 51) * 2
```

It means that group 1 is composed of individuals with ids up to 50, and group 2 with ids from 51. The associated scale parameters are defined by

```
[Scale]
// Group_number  Scale  LowerBound   UpperBound  Status
       1           1        0.001       1000        1
       2           1        0.001       1000        0
```

Clearly, only one of them is identifiable. Note that this allows to avoid complicated tricks based on nested structures in the presence of heterogeneous populations.

More coefficients

05mnl-beta.mod

Here, we have included some additional coefficients into the model.

```
BETA1       0          -10000          10000       0
BETA2       0          -10000          10000       0
```

There are both significant:

```
Name  Value           Robust Std err Robust t-test
BETA1 +7.5924002e-001 +3.7156070e-002 +2.0433808e+001
BETA2 +7.7572746e-001 +3.6835050e-002 +2.1059493e+001
```

But they are correlated in such a way that the hypothesis that they are equal cannot be rejected, at a 95% level:

```
Coefficient1 Coefficient2      Rob. cov.     Rob. corr.    Rob. t-test
~~~~~~~~~~~~ ~~~~~~~~~~~~      ~~~~~~~~~     ~~~~~~~~~~    ~~~~~~~~~~~
BETA1        BETA2        +1.0688450e-03 +7.8095080e-01 -6.7326255e-01 *
```

Modification of an attribute

06mnl-modif-attrib.mod

This example exploits the looping feature on the Expressions section.

```
[Utilities]
// Id Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 +
  1   Alt1   av1   ASC1 * one + BETA1 * logx11 + BETA2 * x12
  2   Alt2   av2   ASC2 * one + BETA1 * logx21 + BETA2 * x22
  3   Alt3   av3   ASC3 * one + BETA1 * logx31 + BETA2 * x32
  4   Alt4   av4   ASC4 * one + BETA1 * logx41 + BETA2 * x42
  5   Alt5   av5   ASC5 * one + BETA1 * logx51 + BETA2 * x52
  6   Alt6   av6   ASC6 * one + BETA1 * logx61 + BETA2 * x62

[Expressions]
// Define here arithmetic expressions for name that are not directly
// available from the data
one = 1

$LOOP{ZZ 1 6 1} logxZZ1 = log( xZZ1 + 15.0 )
```

A simple Mixed Logit model

07mixed-mnl.mod

This example allows for a single normally distributed random coefficient BETA1 [SIGMA1]. More detailed examples are provided in the next section.

07mixed-unif-mnl.mod

This example allows for a single uniformly distributed random coefficient BETA1 { SIGMA1 }.

Note that, in both cases, the number of draws is very low, because the only purpose of these example is to illustrate the syntax, and to test the software. In practice, the number of draws should be at least 1000.

Discrete mixture

08mixed-discrete.mod

This example allows for a the coefficient BETA1 to follow a discrete distribution. It can take two values: B1, with probability W1, and B2, with probability W2. For the model to make sense, it is necessary to impose the linear constraint:

```
[LinearConstraints]
W1 + W2 = 1.0
```

Due to the presence of this constraint, algorithm BIO cannot be used. Therefore, a file 08mixed-discrete.par has been created, where the algorithm DONLP2 is preferred:

```
gevAlgo = "DONLP2"
```

A simple Non Linear model

09mnl-nonlinear.mod

The example 09mnl-nonlinear.mod demonstrates how to add non linear components to the specification of the utilities. In this case we apply a Box-Tuckey transformation to the x variables in the specification of utility 1. The code to implement this is:

```
[Utilities]
// Id Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ... )
  1   Alt1   av1   ASC1 * one
  2   Alt2   av2   ASC2 * one + BETA1 * x21 + BETA2 * x22
  3   Alt3   av3   ASC3 * one + BETA1 * x31 + BETA2 * x32
  4   Alt4   av4   ASC4 * one + BETA1 * x41 + BETA2 * x42
  5   Alt5   av5   ASC5 * one + BETA1 * x51 + BETA2 * x52
  6   Alt6   av6   ASC6 * one + BETA1 * x61 + BETA2 * x62

[GeneralizedUtilities]
1  BETA1 * ((x11 + 10 ) ^ LAMBDA11 - 1) / LAMBDA11
 + BETA2 * ((x12 + 10 ) ^ LAMBDA12 - 1) / LAMBDA12
```

09mnl-nonlinear-deriv.mod

Same file, where the derivatives are explicitly coded in the specification file as follows:

```
[Derivatives]
1 BETA1 ((x11 + 10 ) ^ LAMBDA11 - 1) / LAMBDA11
1 BETA2 ((x12 + 10 ) ^ LAMBDA12 - 1) / LAMBDA12
1 LAMBDA11
      BETA1 * ((x11 + 10) ^ LAMBDA11 * LN(x11 + 10) * LAMBDA11
             - (x11 + 10) ^ LAMBDA11 + 1) / (LAMBDA11 * LAMBDA11 )
1 LAMBDA12
      BETA2 * ((x12 + 10) ^ LAMBDA12 * LN(x12 + 10) * LAMBDA12
             - (x12 + 10) ^ LAMBDA12 + 1) / (LAMBDA12 * LAMBDA12 )
```

A simple Nested Logit model

10nl.mod

A Nested Logit model with two nests, A and B, is tested. Alternatives 1, 2 and 3 belong to nest A, and the other alternatives to nest B.

```
[NLNests]
// Name paramvalue  LowerBound UpperBound  status list of alt
NESTA     1.0           1.0       10.0         0   1 2 3
NESTB     1.0           1.0       10.0         0   4 5 6
```

Note that it is normalized from the top, that is $\mu = 1$, imposing that NESTA ≥ 1 and NESTB ≥ 1, in order to obtain the conditions required by the theory, that is

$$0 \leq \mu/\text{NESTA} \leq 1,$$

and

$$0 \leq \mu/\text{NESTB} \leq 1.$$

10nl-bottom.mod

In the file 10nl-bottom.mod, the exact same model is described, but normalized from the bottom, while the previous was normalized from the top. The coefficient of nest A is constrained to 1, and the μ parameter is estimated.

```
[Mu]
// Value LowerBound UpperBound Status
     1.0        0.0        1.0      0
```

It is important to understand that both models are equivalent, even if the estimated parameters do not have the same values. In the following table, the column "Top" contains the estimated parameters for the model normalized at the top, and "Bottom" the estimated parameters for the model normalized at the bottom. The third column contains the scaled parameters, that is each parameter multiplied by μ, except for the nests parameters, where the value μ/μ_m is reported, m being the nest. These values are independent of the normalization, and should be used when comparing models.

	Top	Bottom	Scaled
ASC1	0	0	0
ASC2	-0.0897	-0.1527	-0.0897
ASC3	-0.0650	-0.1107	-0.0650
ASC4	-0.3282	-0.5587	-0.3282
ASC5	0.5651	0.9619	0.5651
ASC6	-0.8872	-1.5102	-0.8872
BETA1	0.5350	0.9107	0.5350
BETA2	0.5537	0.9426	0.5538
MU	1.0000	0.5875	
NESTA	1.7023	1.0000	0.5874
NESTB	3.2339	1.8996	0.3092

10nl-constrained.mod

In this example, the nest parameters of the nested logit model are constrained to be equal. Note that it is not a normalization constraint, as the drop in loglikelihood from -848.171 to -858.941 illustrates.

```
[LinearConstraints]
NESTA - NESTB = 0.0
```

A Cross-Nested Logit model

11cnl.mod

The CNL model has two nests, A and B, and each alternative belongs to both nests. The α parameters are set to 1/2:

```
[CNLAlpha]
// Alt Nest value LowerBound UpperBound  status
Alt1 NESTA   0.5   0.0001  1.0   1
Alt2 NESTA   0.5   0.0001  1.0   1
Alt3 NESTA   0.5   0.0001  1.0   1
Alt4 NESTA   0.5   0.0001  1.0   1
Alt5 NESTA   0.5   0.0001  1.0   1
Alt6 NESTA   0.5   0.0001  1.0   1
Alt1 NESTB   0.5   0.0001  1.0   1
Alt2 NESTB   0.5   0.0001  1.0   1
Alt3 NESTB   0.5   0.0001  1.0   1
Alt4 NESTB   0.5   0.0001  1.0   1
Alt5 NESTB   0.5   0.0001  1.0   1
Alt6 NESTB   0.5   0.0001  1.0   1
```

12cnl.mod

In the file 12cnl.mod, the α parameters are now estimated. For each alternative, the sum of corresponding α parameters must sum up to 1.0. Therefore, we add the constraints

```
[LinearConstraints]
NESTA_Alt1 + NESTB_Alt1 = 1.0
NESTA_Alt2 + NESTB_Alt2 = 1.0
NESTA_Alt3 + NESTB_Alt3 = 1.0
NESTA_Alt4 + NESTB_Alt4 = 1.0
NESTA_Alt5 + NESTB_Alt5 = 1.0
NESTA_Alt6 + NESTB_Alt6 = 1.0
```

13cnl.mod

The file 13cnl.mod is the same as 12cnl.mod, with another starting point based on the estimates of the nested logit model.

A Network GEV model

14ngev.mod

The Nested Logit model of file 10nl.mod is described here using the Network GEV syntax. Note that the high level of generality provided by the network GEV model comes with a cost. On my computer, a Network GEV model takes much longer to estimate than the same Nested Logit model.

15ngev.mod

It is important to note that the model formulation of the Network GEV model (Daly and Bierlaire, 2006) is not consistent with the Cross-Nested Logit model formulation. In order to illustrate it, the file 15ngev.mod is mimicking the model described in 11cnl.mod, but should be used for syntax considerations only. In general, the two models will give different results, although they are theoretically equivalent, for two reasons.

1. The normalization conditions of the Network GEV models are not clear, and definitely nonlinear. They have not been specified in the file 15ngev.mod, and it complicates the run of the algorithm as the optimization is degenerate.

2. When the α parameters are estimated in a CNL or a Network GEV model, the loglikelihood function exhibits many local maxima. It is very rare that estimations on the two versions of the model leads to the same local maximum.

Panel data

16panel.mod

The MNL model from file 05mnl-beta.mod combined with individual specific error components ZERO [SIGMA] * one, where SIGMA is estimated. Note that ZERO must be declared, but is fixed to 0. The section

```
[PanelData]
Id
ZERO_SIGMA
```

tells Biogeme 2.4 that individuals ids can be found under `Id` in the sample file, and that the random coefficient `ZERO_SIGMA` does not vary across observations from the same individual. Note that it is assumed in Biogeme 2.4 that the data file is sorted in such a way that all observations from each individual are successive in the data file.

Expressions

17expressions.mod

Illustrates the syntax of expressions. Namely, a dummy variable which is defined by

$$\text{dum12} = \begin{cases} 1 & \text{if x11} \geq 1 \text{ or x21} \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

is coded as

```
dum12 = ( x11 >= 1 ) || ( x21 >= 1 )
```

Selection bias

18selectionBias.mod

Biogeme 2.4 has the capability to correct for selection bias in some circumstances. The syntax is simple: the list of additionnal parameters to be estimated must be listed together with their associated alternative in the Section [SelectionBias] of the .mod file, as illustrated in this file. See Section 1.2 and Bierlaire et al., 2008 for more details.

Discrete distributions and panel data

19panelDiscrete.mod

Illustrate a model where one random coefficient has a continuous distribution, and another a discrete distribution, in a panel data context.

Seminonparametric transformation

20legendre.mod

Example of a mixture of MNL (actually, same as `07mixed-mnl.mod`) where a seminonparametric transformation, based on the Legendre polynomials of degree 1, 3 and 4, has been applied. Not that, in practice, it is more common to use consecutive terms. Again, this example is designed to illustrate the syntax, and to emphasize that some terns may be omitted if necessary. See **?** for details.

Lognormal distribution

21mixed-lognormal.mod

Example of a mixture with a lognormally distributed coefficient.

21mixed-lognormal.mod

Same example where the derivatives are explicitly implemented by the user. It illustrate the use of the operator `$DERIV` used to ask BIOGEME to compute the derivative for you. It is mandatory to use it when random variables are involved, such as in this example.

Ordinal logit

22ordinalLogit.mod

Exact same model as in the file `00bl.mod`, but using the syntax of an ordinal logit model (with only two categories, in this case). It is interesting to compare the sign of the coefficients between the two models.

23ordinalLogit.mod

Example of a model specification file for an ordinal logit model with 4 categories. Note the following section, defining the categories:

```
[OrdinalLogit]
1 $NONE    //  Category 1 spans -infty --> tau1
2 tau1     //           2 spans tau1   --> tau2
3 tau2     //           3 spans tau2   --> tau3
4 tau3     //           4 spans tau3   --> +infty
```

Remember the convention introduced in Section 2.14:

- the model must contain exactly two alternatives;
- the identifier of the alternative is completely ignored;
- alternative i is the first by alphabetical order, and alternative j is the second.

The estimation of all basic examples are summarized in Table 2.3.

2.21 Examples of logit kernel (mixed logit) formulations

*T*he examples in this section have been kindly prepared and tested by D. Bolduc and M.-H. Godbout

In this section, we explore several versions of logit kernel formulations in order to demonstrate the capabilities of Biogeme. To achieve this, we generate a sample of 1000 observations arising from a very general model specification which contains, as a special case, several submodels that are often used in applied work.

The data generating process

This subsection introduces the notation, describes the general model and shows how we generated the synthetic sample datafile we provide with the examples. The following section gives the specialized form of the model and provides the associated input files. Let's consider a choice situation involving the choice among 6 alternatives. The following data generating process is used to generate data for a sample of 1000 synthetic choices:

$$
\begin{array}{lllllll}
U_{1n} = & \alpha_1 & +X_{11n}\beta_1 & +X_{12n}\beta_2 & +\sigma_1\xi_{1n} & & +\nu_{1n} \\
U_{2n} = & \alpha_2 & +X_{21n}\beta_1 & +X_{22n}\beta_2 & +\sigma_1\xi_{1n} & +\sigma_2\xi_{2n} & +\nu_{2n} \\
U_{3n} = & \alpha_3 & +X_{31n}\beta_1 & +X_{32n}\beta_2 & & +\sigma_2\xi_{2n} & +\nu_{3n} \\
U_{4n} = & \alpha_4 & +X_{41n}\beta_1 & +X_{42n}\beta_2 & & +\sigma_2\xi_{2n} & +\nu_{4n} \\
U_{5n} = & \alpha_5 & +X_{51n}\beta_1 & +X_{52n}\beta_2 & & +\sigma_2\xi_{2n} \quad +\sigma_3\xi_{3n} & +\nu_{5n} \\
U_{6n} = & & +X_{61n}\beta_1 & +X_{62n}\beta_2 & & +\sigma_3\xi_{3n} & +\nu_{6n}
\end{array}
\tag{2.14}
$$

In this specification, many interesting effects are allowed. For instance, for a given alternative i, heteroscedasticity in utility i could be modeled using a random alternative specific constant (ASC). The second two columns contains two variables X_1 and X_2 associated with two generic coefficients. Flexibility would allow them to be random. These components would permit to model heterogeneity across individuals. In our example, β_1 and β_2 are assumed to come from a joint normal distribution with respective means $\bar{\beta}_1$ and $\bar{\beta}_2$ and variance and covariance matrix:

$$\Sigma_\beta = \begin{bmatrix} var(\beta_1) & cov(\beta_1, \beta_2) \\ cov(\beta_1, \beta_2) & var(\beta_2) \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix}. \tag{2.15}$$

For convenience, the $X'_{kin}s$, $k = 1, 2$, $i = 1, ..., 6$ and $n = 1, ..., N = 1000$, are i.i.d. random variates generated from a standard normal distribution. The ξ_{in} are error component terms (factors) that allow to model the correlation across the utilities. Here, alternatives 1 and 2 are related through factor 1, alternatives 2, 3, 4 and 5 are related through common factor 2 and the last two alternatives have factor 3 in common. The $\nu'_{in}s$ are i.i.d. Gumbel error terms.

On correlated random coefficients

`[Cholesky factorization]` In the above subsection, we assumed that:

$$\begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} \sim MVN(\begin{pmatrix} \bar{\beta}_1 \\ \bar{\beta}_2 \end{pmatrix}, \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix}).$$

Using the Cholesky factorization, the vector $(\beta_1, \beta_2)'$ can be replaced by:

$$\begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} \bar{\beta}_1 \\ \bar{\beta}_2 \end{pmatrix} + \begin{bmatrix} p_{11} & 0 \\ p_{21} & p_{22} \end{bmatrix} \begin{bmatrix} \zeta_1 \\ \zeta_2 \end{bmatrix} \tag{2.16}$$

where ζ_1 and ζ_2 are i.i.d. standard normal variates. Using a matrix notation, we write it as:

$$\beta = \bar{\beta} + P\zeta, \tag{2.17}$$

where P corresponds to a lower triangular Cholesky factorization matrix such that $PP' = \Sigma_\beta$.

We now consider several submodels of the above general formulation.

Linear specification with independent normally distributed random coefficients

In this version, the model is written as:

$$\begin{array}{rcccccc} U_{1n} = & \alpha_1 & +X_{11n}\beta_1 & +X_{21n}\beta_2 & +\nu_{1n} \\ U_{2n} = & \alpha_2 & +X_{12n}\beta_1 & +X_{22n}\beta_2 & +\nu_{2n} \\ U_{3n} = & \alpha_3 & +X_{13n}\beta_1 & +X_{23n}\beta_2 & +\nu_{3n} \\ U_{4n} = & \alpha_4 & +X_{14n}\beta_1 & +X_{24n}\beta_2 & +\nu_{4n} \\ U_{5n} = & \alpha_5 & +X_{15n}\beta_1 & +X_{25n}\beta_2 & +\nu_{5n} \\ U_{6n} = & & +X_{16n}\beta_1 & +X_{26n}\beta_2 & +\nu_{6n} \end{array} \tag{2.18}$$

where the $\alpha's$ are fixed and where the $\beta_k, k = 1, 2$ are independently generated as follows: $\beta_1 \sim N(\bar{\beta}_1, \sigma_1^2)$ and $\beta_2 \sim N(\bar{\beta}_2, \sigma_2^2)$.

Input file

17_2.mod

The main sections of the Biogeme file required to estimate this model is as follows:

```
[Beta]
// Name   Value     LowerBound UpperBound  status (0=variable, 1=fixed)
ASC1        1.0          -10.0     10.0         0
ASC2        1.0          -10.0     10.0         0
ASC3        1.0          -10.0     10.0         0
ASC4        1.0          -10.0     10.0         0
ASC5        1.0          -10.0     10.0         0
BETA1       1.0          -10.0     10.0         0
BETA2       1.0          -10.0     10.0         0
BETA1_S     1.0          -10.0     10.0         0
BETA2_S     1.0          -10.0     10.0         0

[Utilities]
// Id  Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ... )
 1   Alt1  av1  ASC1 * one + BETA1 [ BETA1_S ] * x11 + BETA2 [ BETA2_S ] * x12
 2   Alt2  av2  ASC2 * one + BETA1 [ BETA1_S ] * x21 + BETA2 [ BETA2_S ] * x22
 3   Alt3  av3  ASC3 * one + BETA1 [ BETA1_S ] * x31 + BETA2 [ BETA2_S ] * x32
 4   Alt4  av4  ASC4 * one + BETA1 [ BETA1_S ] * x41 + BETA2 [ BETA2_S ] * x42
 5   Alt5  av5  ASC5 * one + BETA1 [ BETA1_S ] * x51 + BETA2 [ BETA2_S ] * x52
 6   Alt6  av6              BETA1 [ BETA1_S ] * x61 + BETA2 [ BETA2_S ] * x62
```

```
[Model]
// Currently, only $MNL (multinomial logit), $NL (nested logit), $CNL
// (cross-nested logit) and $NGEV (Network GEV model) are valid keywords
//
$MNL
```

In the case with independent random coefficients, the parameter name between the brackets designates the standard deviation of the random parameter term with mean specified on the left of the bracket.

Linear specification with correlated normally distributed random coefficients

In this version, the model is written as:

$$\begin{aligned}
U_{1n} &= \alpha_1 + X_{11n}\beta_1 + X_{21n}\beta_2 + \nu_{1n} \\
U_{2n} &= \alpha_2 + X_{12n}\beta_1 + X_{22n}\beta_2 + \nu_{2n} \\
U_{3n} &= \alpha_3 + X_{13n}\beta_1 + X_{23n}\beta_2 + \nu_{3n} \\
U_{4n} &= \alpha_4 + X_{14n}\beta_1 + X_{24n}\beta_2 + \nu_{4n} \\
U_{5n} &= \alpha_5 + X_{15n}\beta_1 + X_{25n}\beta_2 + \nu_{5n} \\
U_{6n} &= \quad\; + X_{16n}\beta_1 + X_{26n}\beta_2 + \nu_{6n}
\end{aligned} \tag{2.19}$$

where the $\alpha's$ are fixed and where each $\beta_k, k = 1, 2$ are generated from equation (2.16) where the $p's$ are Cholesky terms.

Input file

`17_3.mod`

The main sections of the Biogeme file required to estimate this model is as follows:

```
[Beta]
// Name   Value     LowerBound UpperBound  status (0=variable, 1=fixed)
ASC1        1.0          -10.0     10.0         0
ASC2        1.0          -10.0     10.0         0
ASC3        1.0          -10.0     10.0         0
ASC4        1.0          -10.0     10.0         0
ASC5        1.0          -10.0     10.0         0
BETA1       1.0          -10.0     10.0         0
```

```
BETA2       1.0          -10.0     10.0         0
BETA1_S     1.0          -10.0     10.0         0
BETA2_S     1.0          -10.0     10.0         0

[Utilities]
// Id  Name Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ... )
 1   Alt1   av1 ASC1  * one + BETA1 [ BETA1_S ] * x11 + BETA2 [ BETA2_S ] * x12
 2   Alt2   av2 ASC2  * one + BETA1 [ BETA1_S ] * x21 + BETA2 [ BETA2_S ] * x22
 3   Alt3   av3 ASC3  * one + BETA1 [ BETA1_S ] * x31 + BETA2 [ BETA2_S ] * x32
 4   Alt4   av4 ASC4  * one + BETA1 [ BETA1_S ] * x41 + BETA2 [ BETA2_S ] * x42
 5   Alt5   av5 ASC5  * one + BETA1 [ BETA1_S ] * x51 + BETA2 [ BETA2_S ] * x52
 6   Alt6   av6               BETA1 [ BETA1_S ] * x61 + BETA2 [ BETA2_S ] * x62


[ParameterCovariances]
 //  Par_i         Par_j    Value LowerBound UpperBound status (0=variable, 1=fi
BETA1_BETA1_S BETA2_BETA2_S   1.0      -10.0     10.0        0

[Model]
// Currently, only $MNL (multinomial logit), $NL (nested logit), $CNL
// (cross-nested logit) and $NGEV (Network GEV model) are valid keywords
//
$MNL
```

The only difference with the previous example is that, we added the section ParameterCovariances which indicates which pairs of coefficients are correlated. In a situation with correlation, the coefficients describing the variance covariance structure are Cholesky factorization terms. Thus, BETA1_S corresponds to p_{11}, BETA2_S corresponds to p_{22} and the term called BETA1_BETA1_S_BETA2_BETA2_S in Biogeme corresponds to p_{21}. In the output file, the estimates first presented correspond to those coefficients. Then the Biogeme output converts the Cholesky terms into variances and covariances. By definition, in the case with independent random coefficients, the terms on the diagonal of the diagonal Cholesky factorization matrix directly correspond to the standard deviations of the respective random coefficients.

Specification with correlated normally and lognormally distributed random coefficients

In this version, we allow β_1 to be normally distributed and β_2 to be lognormally distributed. They are also allowed to be correlated. The model is written as:

$$\begin{array}{lllll} U_{1n} = & \alpha_1 & +X_{11n}\beta_1 & +X_{21n}\beta_2 & +\nu_{1n} \\ U_{2n} = & \alpha_2 & +X_{12n}\beta_1 & +X_{22n}\beta_2 & +\nu_{2n} \\ U_{3n} = & \alpha_3 & +X_{13n}\beta_1 & +X_{23n}\beta_2 & +\nu_{3n} \\ U_{4n} = & \alpha_4 & +X_{14n}\beta_1 & +X_{24n}\beta_2 & +\nu_{4n} \\ U_{5n} = & \alpha_5 & +X_{15n}\beta_1 & +X_{25n}\beta_2 & +\nu_{5n} \\ U_{6n} = & & +X_{16n}\beta_1 & +X_{26n}\beta_2 & +\nu_{6n} \end{array} \tag{2.20}$$

where the α's are fixed and where each $\beta_k, k = 1, 2$ are generated from the equation:

$$\begin{pmatrix} \beta_1 \\ \ln\ \beta_2 \end{pmatrix} = \begin{pmatrix} \bar{\beta}_1 \\ \bar{\beta}_2 \end{pmatrix} + \begin{bmatrix} p_{11} & 0 \\ p_{21} & p_{22} \end{bmatrix} \begin{bmatrix} \zeta_1 \\ \zeta_2 \end{bmatrix} \tag{2.21}$$

where the p's are Cholesky terms.

Input file

`17_4.mod`

`[Example LogNormal]` The main sections of the Biogeme file required to estimate this model is as follows:

```
[Beta]
// Name   Value     LowerBound UpperBound  status (0=variable, 1=fixed)
ASC1        1.0          -10.0     10.0         0
ASC2        1.0          -10.0     10.0         0
ASC3        1.0          -10.0     10.0         0
ASC4        1.0          -10.0     10.0         0
ASC5        1.0          -10.0     10.0         0
BETA1       1.0          -10.0     10.0         0
BETA2       1.0          -10.0     10.0         0
BETA1_S     1.0          -10.0     10.0         0
BETA2_S     1.0          -10.0     10.0         0

[Utilities]
// Id  Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ... )
   1   Alt1    av1     ASC1  * one + BETA1 [ BETA1_S ] * x11
   2   Alt2    av2     ASC2  * one + BETA1 [ BETA1_S ] * x21
   3   Alt3    av3     ASC3  * one + BETA1 [ BETA1_S ] * x31
   4   Alt4    av4     ASC4  * one + BETA1 [ BETA1_S ] * x41
   5   Alt5    av5     ASC5  * one + BETA1 [ BETA1_S ] * x51
   6   Alt6    av6                   BETA1 [ BETA1_S ] * x61
```

```
[GeneralizedUtilities]
// Id Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ... )
1  exp( BETA2 [ BETA2_S ] ) * x12
2  exp( BETA2 [ BETA2_S ] ) * x22
3  exp( BETA2 [ BETA2_S ] ) * x32
4  exp( BETA2 [ BETA2_S ] ) * x42
5  exp( BETA2 [ BETA2_S ] ) * x52
6  exp( BETA2 [ BETA2_S ] ) * x62


[ParameterCovariances]
 //  Par_i         Par_j      Value  LowerBound UpperBound  status
 BETA1_BETA1_S BETA2_BETA2_S   1.0      -10.0     10.0        0
```

As mentioned in the manual, non linearities in the parameters must absolutely be incorporated in the GeneralizedUtilities section of the input file.

Linear specification with independent heteroscedastic utilities

In this version, the model is written as:

$$\begin{aligned} U_{1n} &= \alpha_1 + X_{11n}\beta_1 + X_{21n}\beta_2 + \nu_{1n} \\ U_{2n} &= \alpha_2 + X_{12n}\beta_1 + X_{22n}\beta_2 + \nu_{2n} \\ U_{3n} &= \alpha_3 + X_{13n}\beta_1 + X_{23n}\beta_2 + \nu_{3n} \\ U_{4n} &= \alpha_4 + X_{14n}\beta_1 + X_{24n}\beta_2 + \nu_{4n} \\ U_{5n} &= \alpha_5 + X_{15n}\beta_1 + X_{25n}\beta_2 + \nu_{5n} \\ U_{6n} &= \quad + X_{16n}\beta_1 + X_{26n}\beta_2 + \nu_{6n} \end{aligned} \tag{2.22}$$

where the $\alpha's$ are independent random variables each one with its own specific variance, and where each $\beta_k, k = 1, 2$ are fixed coefficients. Identification conditions are discussed in Walker, Ben-Akiva and Bolduc (2004).

Input file

`17_5.mod`

The main sections of the Biogeme file required to estimate this model is as follows:

```
[Beta]
// Name   Value     LowerBound UpperBound  status (0=variable, 1=fixed)
```

```
ASC1        1.0          -10.0     10.0         0
ASC2        1.0          -10.0     10.0         0
ASC3        1.0          -10.0     10.0         0
ASC4        1.0          -10.0     10.0         0
ASC5        1.0          -10.0     10.0         0
BETA1       1.0          -10.0     10.0         0
BETA2       1.0          -10.0     10.0         0
ASC1_S      1.0          -10.0     10.0         0
ASC2_S      1.0          -10.0     10.0         0
ASC3_S      1.0          -10.0     10.0         0
ASC4_S      1.0          -10.0     10.0         0
ASC5_S      1.0          -10.0     10.0         0

[Utilities]
// Id  Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ... )
   1   Alt1    av1     ASC1 [ ASC1_S ]  * one + BETA1  * x11  + BETA2  * x12
   2   Alt2    av2     ASC2 [ ASC2_S ]  * one + BETA1  * x21  + BETA2  * x22
   3   Alt3    av3     ASC3 [ ASC3_S ]  * one + BETA1  * x31  + BETA2  * x32
   4   Alt4    av4     ASC4 [ ASC4_S ]  * one + BETA1  * x41  + BETA2  * x42
   5   Alt5    av5     ASC5 [ ASC5_S ]  * one + BETA1  * x51  + BETA2  * x52
   6   Alt6    av6                              BETA1  * x61  + BETA2  * x62


[Model]
// Currently, only $MNL (multinomial logit), $NL (nested logit), $CNL
// (cross-nested logit) and $NGEV (Network GEV model) are valid keywords
//
$MNL
```

Linear specification with error component structure

In this version, the model is written as:

$$\begin{array}{rcccccccc}
U_{1n} = & \alpha_1 & +X_{11n}\beta_1 & +X_{12n}\beta_2 & +\sigma_1\xi_{1n} & & & +\nu_{1n} \\
U_{2n} = & \alpha_2 & +X_{21n}\beta_1 & +X_{22n}\beta_2 & +\sigma_1\xi_{1n} & +\sigma_2\xi_{2n} & & +\nu_{2n} \\
U_{3n} = & \alpha_3 & +X_{31n}\beta_1 & +X_{32n}\beta_2 & & +\sigma_2\xi_{2n} & & +\nu_{3n} \\
U_{4n} = & \alpha_4 & +X_{41n}\beta_1 & +X_{42n}\beta_2 & & +\sigma_2\xi_{2n} & & +\nu_{4n} \\
U_{5n} = & \alpha_5 & +X_{51n}\beta_1 & +X_{52n}\beta_2 & & +\sigma_2\xi_{2n} & +\sigma_3\xi_{3n} & +\nu_{5n} \\
U_{6n} = & & +X_{61n}\beta_1 & +X_{62n}\beta_2 & & & +\sigma_3\xi_{3n} & +\nu_{6n}
\end{array} \tag{2.23}$$

where the α's and each $\beta_k, k = 1, 2$ are fixed parameters.

Input file

17_6.mod

The main sections of the Biogeme file required to estimate this model is as follows:

```
[Beta]
// Name   Value     LowerBound UpperBound  status (0=variable, 1=fixed)
ASC1        1.0          -10.0     10.0         0
ASC2        1.0          -10.0     10.0         0
ASC3        1.0          -10.0     10.0         0
ASC4        1.0          -10.0     10.0         0
ASC5        1.0          -10.0     10.0         0
BETA1       1.0          -10.0     10.0         0
BETA2       1.0          -10.0     10.0         0
fact1       0.0          -10.0     10.0         1
fact2       0.0          -10.0     10.0         1
fact3       0.0          -10.0     10.0         1
fact1_s     1.0          -10.0     10.0         0
fact2_s     1.0          -10.0     10.0         0
fact3_s     1.0          -10.0     10.0         0

[Utilities]
// Id  Name  Avail  linear-in-parameter expression (beta1*x1 + beta2*x2 + ... )
   1   Alt1    av1     ASC1 * one + BETA1   * x11  + BETA2  * x12 + fact1 [ fact
   2   Alt2    av2     ASC2 * one + BETA1   * x21  + BETA2  * x22 + fact1 [ fact
                       + fact2 [ fact2_s ] * one
   3   Alt3    av3     ASC3 * one + BETA1   * x31  + BETA2  * x32 + fact2 [ fact
   4   Alt4    av4     ASC4 * one + BETA1   * x41  + BETA2  * x42 + fact2 [ fact
   5   Alt5    av5     ASC5 * one + BETA1   * x51  + BETA2  * x52 + fact2 [ fact
                     + fact3 [ fact3_s ] * one
   6   Alt6    av6                  BETA1   * x61  + BETA2  * x62 + fact3 [ fact


[Model]
// Currently, only $MNL (multinomial logit), $NL (nested logit), $CNL
// (cross-nested logit) and $NGEV (Network GEV model) are valid keywords
//
$MNL
```

Specification with non linear random heterogeneous value of time

For this specialized version of the model, we consider estimating the following model:

$$\begin{array}{rcl} U_{1n} = & \alpha_1 & +X_{11n}\beta_1 \quad +(cost_{1n} \cdot inc_n^{\eta})\beta_2 \quad +\nu_{1n} \\ U_{2n} = & \alpha_2 & +X_{21n}\beta_1 \quad +(cost_{2n} \cdot inc_n^{\eta})\beta_2 \quad +\nu_{2n} \\ U_{3n} = & \alpha_3 & +X_{31n}\beta_1 \quad +(cost_{3n} \cdot inc_n^{\eta})\beta_2 \quad +\nu_{3n} \\ U_{4n} = & \alpha_4 & +X_{41n}\beta_1 \quad +(cost_{4n} \cdot inc_n^{\eta})\beta_2 \quad +\nu_{4n} \\ U_{5n} = & \alpha_5 & +X_{51n}\beta_1 \quad +(cost_{5n} \cdot inc_n^{\eta})\beta_2 \quad +\nu_{5n} \\ U_{6n} = & & +X_{61n}\beta_1 \quad +(cost_{6n} \cdot inc_n^{\eta})\beta_2 \quad +\nu_{6n} \end{array} \tag{2.24}$$

where the $\alpha's$ are fixed and where each $\beta_k, k = 1, 2$ are independently generated as follows: $\beta_1 \sim N(\bar{\beta}_1, \sigma_1^2)$ and $\beta_2 \sim N(\bar{\beta}_2, \sigma_2^2)$. The model contains a cost variable which vary across alternatives and an income variable which is evaluated to the power η where η is a normally distributed random coefficient. The database may be found in the directory where the Biogeme input file resides.

Input file

`17_7.mod`

The main sections of the Biogeme file required to estimate this model is as follows:

```
[Beta]
// Name   Value  LowerBound UpperBound  status (0=variable, 1=fixed)
ASC1        1.0        -10.0     10.0         0
ASC2        1.0        -10.0     10.0         0
ASC3        1.0        -10.0     10.0         0
ASC4        1.0        -10.0     10.0         0
ASC5        1.0        -10.0     10.0         0
BETA1       1.0        -10.0     10.0         0
BETA2       1.0        -10.0     10.0         0
BETA1_S     1.0        -10.0     10.0         0
BETA2_S     1.0        -10.0     10.0         0
ETA         1.0        -10.0     10.0         0
ETA_S       1.0        -10.0     10.0         0

[Utilities]
```

```
// Id  Name  Avail  linear-in-parameter expression
   1   Alt1    av1     ASC1  * one + BETA1 [ BETA1_S ] * x11
   2   Alt2    av2     ASC2  * one + BETA1 [ BETA1_S ] * x21
   3   Alt3    av3     ASC3  * one + BETA1 [ BETA1_S ] * x31
   4   Alt4    av4     ASC4  * one + BETA1 [ BETA1_S ] * x41
   5   Alt5    av5     ASC5  * one + BETA1 [ BETA1_S ] * x51
   6   Alt6    av6                   BETA1 [ BETA1_S ] * x61

[GeneralizedUtilities]
// Id Name  Avail  linear-in-parameter expression
1  BETA2 [ BETA2_S ] * ( inc ^ ETA [ ETA_S ] ) * cost1
2  BETA2 [ BETA2_S ] * ( inc ^ ETA [ ETA_S ] ) * cost2
3  BETA2 [ BETA2_S ] * ( inc ^ ETA [ ETA_S ] ) * cost3
4  BETA2 [ BETA2_S ] * ( inc ^ ETA [ ETA_S ] ) * cost4
5  BETA2 [ BETA2_S ] * ( inc ^ ETA [ ETA_S ] ) * cost5
6  BETA2 [ BETA2_S ] * ( inc ^ ETA [ ETA_S ] ) * cost6


[Model]
// Currently, only $MNL (multinomial logit), $NL (nested logit), $CNL
// (cross-nested logit) and $NGEV (Network GEV model) are valid keywords
$MNL
```

Again, non linearities must be incorporated in the GeneralizedUtilities section of the input file.

Panel data

In Biogeme, when panel data is used, it is possible to identify which random parameters should vary only across individuals, and not across observations. This functionality is illustrated by the files `17_8.mod` and `17_8simple.mod`.

2.22 Compiling from the sources

Biogeme 2.4 has been developed using GNU C++ (see, among many others, **?**). In general, this environment is available on computer running linux or Mac OS X operating systems. On computers running Windows, several distributions of GNU tools are available (**?**). BIOGEME is actually developed within the `cygwin` environment available from `www.cygwin.com`. The executable itself is compiled within the `Minimalist GNU for Windows (MinGW)` environment, available at `www.mingw.org`. The following assumes that you

are familiar with one of these environments. I have included screen shots from the MinGW environment.

Before starting the installation, make sure to install the Fast Light Toolkit from www.fltk.org in order to be able to compile the Graphical User Interface.

1. First, the distribution must be unzipped in a directory. It is good practice to call that directory biogeme, but it is not required. In this example, the directory is /home/biogeme. After unzipping, the following should appear in the directory:

2. Before compiling, the environment variable MAKEHOME must contain the name of the directory where the source files have been installed. Depending on the Shell you are using, the syntax may be slightly different. If you use tcsh as a shell, type

   ```
   setenv MAKEHOME `pwd`
   ```

   If you use bash as a shell, type

   ```
   export MAKEHOME=`pwd`
   ```

   as illustrated here:

3. Type make or gmake to launch the compilation. Depending on your computer, it may take several minutes. You will probably obtain the following error message:

   Just retype make or gmake. When it is done, you'll obtain something like:

   These executables are now available:

```
biogeme
bioroute/bioroute
biosim/biosim
fltk/winbiogeme.exe
```

Make sure that you move them to a directory in your search path for executables, or that you update your path so that they can be found.

2.23 Parallel computing with BIOGEME

It is possible to run Biogeme 2.4 much faster for the estimation of multinomial logit (MNL) models, or mixtures of MNL, in particular those involving nonlinear utility functions. This is done within the directory `fastbiogeme`. In order to illustrate the process, we estimate the model from the file `16panel.mod` on a linux machine with 8 processors. We first copy the model specification file as well as the data file in the `fastbiogeme` directory.

The core of Biogeme 2.4 is available in this directory within the file `libbiogeme.dll`. If you are using a linux machine, the extension is expected to be `.so`, not `.dll`. In this case, just rename the file.

Now, the idea is to build an executable dedicated to your model. Before doing that, it is useful to define the number of processors on your computer. Edit the file `default.par` and set the parameter `gevNumberOfThreads` to the appropriate value. If you define more processors than the actual number, the program will run just fine, but may be slowed down due to unnecessary overhead. If you only have one processor, it is still worth using this procedure, as the running time is significantly decreased, especially when the model involves nonlinear utility functions. In this case, set the parameter `gevNumberOfThreads` to 1.

When you're done, compile the software using the command

```
make mymodel.exe
```

as illustrated below:

As you can see, this involves the compilation of a C++ code. Consequently, it is recommended to use this feature after you've compiled Biogeme 2.4 on your own computer using the procedure described in Section 2.22. The new executable can now be used instead of `biogeme` using the following command:

```
16panel.exe 16panel sample.dat
```

instead of

```
biogeme.exe 16panel sample.dat
```

This feature is recent. Although we have tested and used it in many circumstances, there may be some problems on some computers. Therefore, it is always good practice to check on simple models that the results provided by the parallel version are the same as those provided by the regular version of Biogeme 2.4. Please report any problem to the users' group.

To illustrate the gain in speed, we have estimated the model `21mixed-lognormal` with 1000 and 5000 draws. The run time for estimation are reported in Table 2.4.

Internal note: Talk about missing values

Model	**L**oglike	**O**bs.	**I**nd.	**B**ETA1	**B**ETA2
00bl	-151.954	1019	1019	0.921	0.954
00bp	-151.709	1019	1019	0.509	0.528
01mnl-basic	-2477.245	1999	1999	-	-
02mnl-weights	-2474.038	1999	1999	-	-
03mnl-weights	-2488.234	1999	1999	-	-
04mnl-heterosample	-2477.241	1999	1999	-	-
05mnl-beta	-891.225	1999	1999	0.779	0.81
06mnl-modif-attrib	-904.776	1999	1999	12	0.793
07mixed-mnl	-890.626[1]	1999	1999	0.782	0.811
07mixed-unif-mnl	-890.823[1]	1999	1999	0.781	0.81
08mixed-discrete	-891.225	1999	1999	-dist.-	0.81
09mnl-nonlinear-deriv	-890.775	1999	1999	0.787	0.809
09mnl-nonlinear	-890.775	1999	1999	0.787	0.809
10nl-bottom	-848.171	1999	1999	0.911	0.943
10nl-constrained	-858.941	1999	1999	0.536	0.558
10nl	-848.171	1999	1999	0.535	0.554
10nlsim	-848.171	1999	1999	0.535	0.554
11cnl	-890.902	1999	1999	0.659	0.685
12cnl	-846.955[2]	1999	1999	0.513	0.53
13cnl	-846.955	1999	1999	0.513	0.53
14ngev	-848.171	1999	1999	0.535	0.554
15ngev	-890.92	1999	1999	0.647	0.672
16panel	-890.008[1]	1999	200	0.782	0.813
17expressions	-891.162	1999	1999	0.776	0.81
18selectionBias	-846.778	1999	1999	0.539	0.558
19panelDiscrete	-890.008[1]	1999	200	-dist.-	0.813
20legendre	-891.34[1]	1999	200	0.791	0.815
21mixed-lognormal-deriv	-895.748	1999	1999	-0.256	0.806
21mixed-lognormal	-895.748	1999	1999	-0.256	0.806
22ordinalLogit	-151.954	1019	1019	-0.921	-0.954
23ordinalLogit	-2113.617	1672	1672	-0.0856	-0.0968

[1] The value will vary from machine to machine due to the random seed

[2] On some machines, 12cnl converges to a loglikelihood of -891.225

Table 2.3: Summary of the estimation of the basic examples

Version	# processors	Draws	
		1000	5000
BIOGEME	1	08:03	38:41
FASTBIOGEME	1	01:37	07:48
	2	01:08	05:43
	4	00:44	03:33
	8	00:31	02:21

Table 2.4: Actual run time (mm:ss) of BIOGEME and FASTBIOGEME

Chapter 3

Pythonbiogeme

3.1 Acknowledgments

I would like to thank the following persons who played various roles in the development of Biogeme along the years. The list is probably not complete, and I apologize for those who are omitted: Alexandre Alahi, Nicolas Antille, Gianluca Antonini, Kay Axhausen, John Bates, Denis Bolduc, David Bunch, Andrew Daly, Anna Fernandez Antolin, Mamy Fetiarison, Mogens Fosgerau, Emma Frejinger, Carmine Gioia, Marie-Hélène Godbout, Stephane Hess, Richard Hurni, Jasper Knockaert, Xinjun Lai, Carolina Osorio, Thomas Robin, Pascal Scheiben, Matteo Sorci, Michaël Thémans, Joan Walker.

Finally, I would like to express a special thank to Moshe Ben-Akiva and Daniel McFadden for their friendship, and for the immense influence that they had and still have on my work.

Bibliography

Abbe, E., Bierlaire, M. and Toledo, T. (2007). Normalization and correlation of cross-nested logit models, *Transportation Research Part B* **4**1(7): 795–808.

Ben-Akiva, M. and Bierlaire, M. (2003). Discrete choice models with applications to departure time and route choice, *in* R. Hall (ed.), *Handbook of Transportation Science, 2nd edition*, Operations Research and Management Science, Kluwer, pp. 7–38. ISBN:1-4020-7246-5.

Ben-Akiva, M., Bolduc, D. and Bradley, M. (1993). Estimation of travel model choice models with randomly distributed values of time, *Transportation Research Record* **1**413: 88–97.

Ben-Akiva, M. E. (1973). *Structure of Passenger Travel Demand Models*, PhD thesis, Massachusetts Institute of Technology.

Ben-Akiva, M. E. and Lerman, S. R. (1985). *Discrete Choice Analysis: Theory and Application to Travel Demand*, MIT Press, Cambridge, Ma.

Berndt, E. K., Hall, B. H., Hall, R. E. and Hausman, J. A. (1974). Estimation and inference in nonlinear structural models, *Annals of Economic and Social Measurement* **3**/4: 653–665.

Bierlaire, M. (2002). The network GEV model, *Proceedings of the 2nd Swiss Transportation Research Conference*, Ascona, Switzerland.

Bierlaire, M. (2006). A theoretical analysis of the cross-nested logit model, *Annals of operations research* **1**44(1): 287–300.

Bierlaire, M., Axhausen, K. and Abay, G. (2001). The acceptance of modal innovation: The case of swissmetro, *Proceedings of the Swiss Transport Research Conference*, Ascona, Switzerland.

Bierlaire, M., Bolduc, D. and McFadden, D. (2008). The estimation of generalized extreme value models from choice-based samples, *Transportation Research Part B* **4**2(4): 381–394.

Conn, A. R., Gould, N. I. M. and Toint, P. (2000). *Trust region methods*, MPS–SIAM Series on Optimization, SIAM.

Daly, A. (1987). Estimating "tree" logit models, *Transportation Research Part B* **2**1(4): 251–268.

Daly, A. (2001). Recursive nested EV model, *ITS Working Paper 559*, Institute for Transport Studies, University of Leeds.

Daly, A. and Bierlaire, M. (2006). A general and operational representation of generalised extreme value models, *Transportation Research Part B* **4**0(4): 285–305.

Dennis, J. E. and Schnabel, R. B. (1996). *Numerical methods for unconstrained optimization and nonlinear equations*, Society for Industrial and Applied Mathematics (SIAM).

Hess, S., Train, K. and Polak, J. (2006). On the use of modified latin hypercube sampling (MLHS) method in the estimation of mixed logit model for vehicle choice, *Transportation Research Part B* **4**0(2): 147–163.

Kauermann, G. and Carroll, R. (2001). A note on the efficiency of sandwich covariance matrix estimation, *Journal of the American Statistical Association* **9**6(456).

McFadden, D. (1978). Modelling the choice of residential location, *in* A. Karlquist *et al.* (ed.), *Spatial interaction theory and residential location*, North-Holland, Amsterdam, pp. 75–96.

Papola, A. (2004). Some developments on the cross-nested logit model, *Transportation Research B* **3**8(9): 833–851.

Small, K. (1987). A discrete choice model for ordered alternatives, *Econometrica* **5**5(2): 409–424.

Train, K. (2009). *Discrete Choice Methods with Simulation*, 2nd edn, Cambridge University Press. http://emlab.berkeley.edu/books/choice2.html.

Vovsha, P. (1997). Cross-nested logit model: an application to mode choice in the Tel-Aviv metropolitan area, *Transportation Research Record* **1**607: 6–15.

Wen, C.-H. and Koppelman, F. S. (2001). The generalized nested logit model, *Transportation Research Part B* **3**5(7): 627–641.

White, H. (1982). Maximum likelihood estimation of misspecified models, *Econometrica* **5**0: 1–25.

Index