Estimating hybrid choice models with Biogeme

Michel Bierlaire Moshe Ben-Akiva Joan Walker

December 24, 2025

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

Series on Biogeme

Contents

The package Biogeme (`biogeme.epfl.ch`) is designed to estimate the parameters of various models using maximum likelihood estimation. It is particularly designed for discrete choice models. In this document, we present how to estimate choice models involving latent variables: hybrid choice models.

We assume that the reader is already familiar with discrete choice models, and has successfully installed Biogeme. This document has been written using Biogeme 3.3.2.

1 Models and notations

The literature on discrete choice models with latent variables is vast (Walker, 2001, Ashok et al., 2002, Greene and Hensher, 2003, Ben-Akiva et al., 2002, to cite just a few). We start this document by a short introduction to the models and the notations.

1.1 Structural equations

A *latent variable* is a variable that cannot be directly observed. It is typically modeled using a **structural equation**, which expresses the latent variable as a function of observed (explanatory) variables and an error term. A general form of such a structural equation is:

$$x^*_{nk} = x^*(x_n; \psi_k) + \omega_{nk}, \tag{1}$$

where n indexes individuals, x^*_{nk} is the kth latent variable of interest, x_n is a vector of observed explanatory variables, ψ_k is a vector of parameters to be estimated, and ω_{nk} is a stochastic error term.

A common specification assumes a linear functional form with i.i.d. normally distributed error terms:

$$x^*_{nk} = \sum_s \psi_{sk} x_{ns} + \sigma_{\omega k}\omega_{nk}, \tag{2}$$

where $\omega_{nk} \sim N(0, 1)$, and $\sigma_{\omega k}$ is a scaling parameter for the error term.

Information about latent variables is obtained indirectly through *measurements*, which are observable manifestations of the underlying latent constructs. For example, in discrete choice models, utility is not directly observed but is inferred from the choices individuals make. The relationship between a latent variable and its associated measurements is described by **measurement equations**. The specific form of these equations depends on the nature of the observed measurements (e.g., continuous, or ordinal).

1.2 Measurement equations: the continuous case

Since latent variables cannot be directly observed, analysts rely on indirect measurements to infer their values. A common approach involves asking respondents to rate the perceived magnitude of the latent construct on an arbitrary scale. For example: *"How would you rate the level of pain that you are experiencing, from 0 (no pain) to 10 (worst pain imaginable)?"*

Each such rating is referred to as an *indicator*, indexed by $\ell = 1, \ldots, L_n$, and is modeled using a **measurement equation**. This equation relates the observed indicator to the latent variables and, sometimes, other explanatory variables:

$$I_{n\ell} = I_\ell(x_n, x^*_n; \lambda_\ell) + \upsilon_{n\ell}, \ \forall \ell = 1, \ldots, L_n, \forall n, \tag{3}$$

where $I_{n\ell}$ denotes the response provided by individual n for indicator ℓ, x^*_n is the latent variable of interest (e.g., pain perception), x_n is a vector of observed explanatory variables (such as socio-demographic characteristics), λ_ℓ is a vector of parameters to be estimated, and $\upsilon_{n\ell}$ is the random error term.

A common specification of the measurement function assumes linearity and i.i.d. normally distributed errors:

$$I_{n\ell} = \lambda_{\ell 0} + \sum_k \lambda_{\ell k} x^*_{nk} + \sigma_{\upsilon\ell}\upsilon_{n\ell}, \quad \forall \ell, \tag{4}$$

where $\lambda_{\ell k}$ are unknown parameters to be estimated, $\sigma_{\upsilon\ell}$ is an indicator-specific scale parameter, and $\upsilon_{n\ell} \sim N(0, 1)$.

If we observe a vector of continuous indicators $I_n = (I_{n1}, \ldots, I_{nL_n})$ for individual n, the contribution to the likelihood function, *conditional on the latent variables* x^*_n, is given by the product:

$$\prod_{\ell=1}^{L_n} \phi\left(\frac{I_{n\ell} - \lambda_{\ell 0} - \sum_k \lambda_{\ell k} x^*_{nk}}{\sigma_{\upsilon\ell}}\right), \tag{5}$$

where $\phi(\cdot)$ denotes the probability density function (pdf) of the standard normal distribution.

If other types of observations are available for the same individual (such as discrete choices), the corresponding components of the likelihood can be multiplied with the expression above. Once all relevant components are combined, the latent variables must be integrated out, as discussed later.

If the continuous indicators are the only data available for individual n, the contribution to the unconditional likelihood becomes:

$$\int_{x^*_n} \left[\prod_{\ell=1}^{L_n} \phi\left(\frac{I_{n\ell} - \lambda_{\ell 0} - \sum_k \lambda_{\ell k} x^*_{nk}}{\sigma_{\upsilon\ell}}\right)\right] f(x^*_n)\, dx^*_n, \tag{6}$$

where $f(x_n^*)$ is the pdf of the vector of latent variables x_n^*. As this integral does not have a closed-form expression, it is approximated using Monte Carlo integration (see Bierlaire, 2019 for a discussion about performing Monte-Carlo integration with Biogeme).

1.3 Measurement equation: the ordinal case

Another type of indicator arises when respondents are asked to evaluate a statement using an ordinal scale. A typical context for this type of measurement is the use of a Likert scale (Likert, 1932), where individuals express their degree of agreement or disagreement with a given statement. For example:

> *"I believe that my own actions have an impact on the planet."*
> Response options: strongly agree (2), agree (1), neutral (0), disagree (−1), strongly disagree (−2).

To model these types of indicators, we represent the observed measurement as an *ordered discrete variable* $I_{n\ell}$, which takes values in a finite, ordered set $\{j_1, j_2, \ldots, j_{M_\ell}\}$. The measurement equation involves two stages:

Step 1: Latent response formulation. We first define a continuous response variable, as explained in Section 1.2, except that it happens to be unobserved (latent) in this case:

$$I^*_{n\ell} = I^*_\ell(x_n, x_n^*; \lambda_\ell) + \upsilon_{n\ell}, \tag{7}$$

where $I^*_{n\ell}$ is a continuous latent variable underlying the reported response, x_n^* is a vector of relevant latent variables (e.g., environmental concern), x_n is a vector of observed explanatory variables (e.g., age, income), λ is a vector of parameters to be estimated, and $\upsilon_{n\ell}$ is a random error term.

Step 2: Discretization via thresholds. Since $I^*_{n\ell}$ is not observed, we relate it to the reported discrete measurement $I_{n\ell}$ through a set of threshold parameters:

$$I_{n\ell} = \begin{cases} j_1 & \text{if } I^*_{n\ell} < \tau_1, \\ j_2 & \text{if } \tau_1 \leqslant I^*_{n\ell} < \tau_2, \\ \vdots & \\ j_m & \text{if } \tau_{m-1} \leqslant I^*_{n\ell} < \tau_m, \\ \vdots & \\ j_M & \text{if } \tau_{M_\ell - 1} \leqslant I^*_{n\ell}, \end{cases} \tag{8}$$

where $\tau_1, \dots, \tau_{M_\ell - 1}$ are threshold parameters to be estimated, satisfying the ordering constraint:

$$\tau_1 \leqslant \tau_2 \leqslant \cdots \leqslant \tau_{M_\ell - 1}. \tag{9}$$

Note that it is customary to use the same set of parameters for all individuals n and all indicators ℓ, which explains the absence of these indices on the parameter τ.

Defining $\tau_0 = -\infty$ and $\tau_{M_\ell} = +\infty$, it simplifies to

$$I_{n\ell} = j_m \text{ if } \tau_{m-1} \leqslant I^*_{n\ell} < \tau_m, \; m = 1, \dots, M_\ell. \tag{10}$$

It is often advantageous to impose a symmetric structure on the definition of the thresholds. In addition, it is more convenient from an estimation standpoint to parameterize the thresholds in terms of differences and to constrain these differences to be positive. For example, when $M_\ell = 4$, the thresholds can be defined as follows:

$$\begin{aligned} \tau_1 &= -\delta_1 - \delta_2, \\ \tau_2 &= -\delta_1, \\ \tau_3 &= \delta_1, \\ \tau_4 &= \delta_1 + \delta_2, \end{aligned}$$

where $\delta_1 > 0$ and $\delta_2 > 0$ are the parameters to be estimated. This parameterization guarantees that the thresholds are strictly ordered and symmetrically centered around zero, which facilitates both identification and interpretation.

If we consider a linear specification,

$$I^*_{n\ell} = \lambda_{\ell 0} + \sum_k \lambda_{\ell k} x^*_{nk} + \sigma_{\upsilon\ell} \upsilon_{n\ell}, \quad \forall \ell, \tag{11}$$

where the error term $\upsilon_{n\ell} \sim N(0, 1)$, the contribution of each indicator ℓ for each observation n to the likelihood function, *conditional on the latent* *variables*, is defined as follows:

$$\begin{aligned}\Pr(I_{n\ell}=j_m|x^*_n,x_n;\lambda_\ell,\Sigma_{\upsilon\ell}) =& \Pr(\tau_{m-1}\leqslant I^*_{n\ell}\leqslant\tau_m)\\ =& \Pr(I^*_{n\ell}\leqslant\tau_m)-\Pr(I^*_{n\ell}\leqslant\tau_{m-1}),\\ =& \Pr\left(\upsilon_{n\ell}\leqslant\frac{\tau_m-\lambda_{\ell 0}-\sum_k\lambda_{\ell k}x^*_{nk}}{\sigma_{\upsilon\ell}}\right)\\ &-\Pr\left(\upsilon_{n\ell}\leqslant\frac{\tau_{m-1}-\lambda_{\ell 0}-\sum_k\lambda_{\ell k}x^*_{nk}}{\sigma_{\upsilon\ell}}\right),\\ =&\Phi\left(\frac{\tau_m-\lambda_{\ell 0}-\sum_k\lambda_{\ell k}x^*_{nk}}{\sigma_{\upsilon\ell}}\right)\\ &-\Phi\left(\frac{\tau_{m-1}-\lambda_{\ell 0}-\sum_k\lambda_{\ell k}x^*_{nk}}{\sigma_{\upsilon\ell}}\right)\end{aligned}\tag{12}$$

where j_m is the observed category for respondent n and indicator ℓ.

This specification is known as the *ordered probit model* and is widely used for modeling ordinal responses that depend on latent constructs.

If we observe a vector of continuous indicators $I_n=(I_{n1},\ldots,I_{nL_n})$ for individual n, the contribution to the likelihood function, *conditional on the latent variables* x^*_n, is given by:

$$\prod_{\ell=1}^{L_n}\Pr(I_{n\ell}=j_m|x^*_n,x_n;\lambda_\ell,\Sigma_{\upsilon\ell}).\tag{13}$$

As in the continous case, if other types of observations are available for the same individual (such as choices), the corresponding components of the likelihood can be multiplied with the expression above. Once all relevant components are combined, the latent variables must be integrated out, as discussed later.

If the continuous indicators are the only data available for individual n, the contribution to the unconditional likelihood becomes:

$$\int_{x^*_n}\left[\prod_{\ell=1}^{L_n}\Pr(I_{n\ell}=j_m|x^*_n,x_n;\lambda_\ell,\Sigma_{\upsilon\ell})\right]f(x^*_n)\,dx^*_n,\tag{14}$$

where $f(x^*_n)$ is the pdf of the vector of latent variables x^*_n. Again, this integral is approximated using Monte-Carlo integration.

1.4 Normalization and identification in latent-variable models

Models with latent variables involve parameters that are not directly identified from the data without additional restrictions. These restrictions, known as *normalizations*, are not substantive assumptions about behavior; rather, they fix the arbitrary units of measurement (location and scale) that are inherent to latent constructs. This section explains *why* normalization is needed, *where* non-identification arises in our notation, and provides practical guidelines. We emphasize the *reference-indicator* strategy because it yields parameters with a direct interpretation and tends to be numerically stable, although it is not the only valid approach.

Latent-variable models are invariant to certain transformations of the latent variables and associated parameters. Without normalization, multiple parameter vectors generate exactly the same probability of the observed data, hence the same likelihood. In such a case, maximization is ill-posed (flat directions), standard errors are meaningless, and numerical optimization or Bayesian sampling may become unstable.

Two systematic sources of non-identification arise:

- **Location (translation) invariance**: the origin of a latent variable is arbitrary.
- **Scale invariance**: the unit (measurement scale) of a latent variable is arbitrary.

Both must be addressed to obtain a well-identified model.

We first consider the continuous measurement equation (4):

$$I_{n\ell} = \lambda_{\ell 0} + \sum_{k} \lambda_{\ell k} x^*_{nk} + \sigma_{\upsilon \ell} \upsilon_{n\ell}, \qquad \upsilon_{n\ell} \sim \mathcal{N}(0,1).$$

The structural equation is (2):

$$x^*_{nk} = \sum_{s} \psi_{sk} x_{ns} + \sigma_{\omega k} \omega_{nk}, \qquad \omega_{nk} \sim \mathcal{N}(0,1).$$

Suppose that the intercept $\lambda_{\ell 0}$ is estimated for at least one indicator that loads on latent variable k. Then, for any constant c_k, define

$$x^{*'}_{nk} = x^*_{nk} + c_k.$$

In the measurement equation, the term $\lambda_{\ell k} x^*_{nk}$ can be rewritten as $\lambda_{\ell k}(x^{*'}_{nk} - c_k)$, which is the same as keeping $x^{*'}_{nk}$ but shifting the intercept:

$$\lambda'_{\ell 0} = \lambda_{\ell 0} - \lambda_{\ell k} c_k.$$

Therefore, the distribution of $I_{n\ell}$ (and hence the likelihood) is unchanged: the model cannot distinguish a shift in the latent variable from a compensating shift in intercepts. This is why the *origin* of each latent variable must be fixed.

Also, for any nonzero constant a_k, define

$$x^{*'}_{nk} = a_k x^*_{nk}.$$

Then $\lambda_{\ell k} x^*_{nk} = (\lambda_{\ell k}/a_k) x^{*'}_{nk}$. Hence, scaling the latent variable can be compensated by inversely scaling all associated loadings. On the structural side, scaling x^*_{nk} can be compensated by scaling $\sigma_{\omega k}$ and the coefficients ψ_{sk}. Again, the likelihood is unchanged: the model cannot infer the *unit* in which x^*_{nk} is measured unless we fix it.

Consequently, for each latent variable k, the analyst must impose *one* location normalization and *one* scale normalization. These two restrictions remove two degrees of freedom per latent variable: one for translation, one for rescaling.

A practical and interpretable approach is to choose, for each latent variable k, a *reference indicator* $\ell(k)$ among the indicators intended to measure x^*_{nk}. The key idea is to anchor the latent variable to a concrete observed scale.

First, we fix the location by fixing one intercept. For each k, impose

$$\boxed{\lambda_{\ell(k)0} = 0.}$$

Because $\lambda_{\ell 0}$ can absorb any shift of x^*_{nk}, setting the intercept of one measurement equation to zero declares where the latent variable "zero" is, in a way that is tied to an observed indicator.

Second, fix the scale by fixing one loading. For each k, impose

$$\boxed{\lambda_{\ell(k)k} = 1.}$$

Because the product $\lambda_{\ell k} x^*_{nk}$ is all that matters, fixing $\lambda_{\ell(k)k} = 1$ defines the unit of x^*_{nk} as the unit that makes the reference indicator respond one-for-one (in its systematic part) to changes in the latent variable.

Importantly, fixing the loading to $+1$ is a *convention.* Since the sign of a latent variable is itself arbitrary, it may be equally appropriate to fix $\lambda_{\ell(k)k} = -1$. This choice simply reverses the orientation of the latent scale, without affecting the likelihood or the model fit. The advantage of using -1 arises when the natural interpretation of the reference indicator runs opposite to the intended interpretation of the latent construct.

For instance, suppose x^*_{nk} represents *environmental concern*, with higher values intended to mean *stronger concern.* Consider a reference indicator based on agreement with the statement "I do not care about the environmental impact of my travel," coded so that higher responses correspond to stronger agreement. In this case, higher values of the indicator imply *lower* environmental concern. Fixing $\lambda_{\ell(k)k} = +1$ would force increases in x^*_{nk} to increase the indicator, leading to a latent variable whose numerical interpretation runs counter to its conceptual meaning. By fixing $\lambda_{\ell(k)k} = -1$ instead, higher values of x^*_{nk} decrease the expected indicator response, restoring a coherent and intuitive interpretation: larger x^*_{nk} corresponds to stronger environmental concern.

Thus, choosing $\lambda_{\ell(k)k} = 1$ or -1 does not affect identification or statistical fit, but it greatly improves the semantic consistency and interpretability of the model parameters.

With $\lambda_{\ell(k)0} = 0$ and $\lambda_{\ell(k)k} = 1$ fixed, it is meaningful to estimate $\sigma_{\upsilon\ell(k)}$: it quantifies the amount of noise in that indicator relative to the anchored latent scale. In other words, it is now a genuine signal-to-noise parameter rather than a normalization device.

If the model contains multiple latent variables, the reference indicator for k should ideally load only on k:

$$\lambda_{\ell(k)h} = 0 \quad \text{for } h \neq k.$$

Inded, if the reference indicator mixes several latent variables, then fixing a single loading to one does not define a clean unit for a single construct; it anchors a linear combination instead, which complicates interpretation and may reintroduce weak identification.

We now consider ordinal indicators (Likert-type) modeled through an ordered probit mechanism (Section 1.3). The essential difference with the continuous case is that the observed indicator $I_{n\ell}$ is not the latent response $I^*_{n\ell}$ itself, but only the interval in which it lies:

$$I_{n\ell} = j_m \quad \text{if} \quad \tau_{m-1} \leqslant I^*_{n\ell} < \tau_m, \qquad m = 1, \ldots, M_\ell,$$

with $\tau_0 = -\infty$ and $\tau_{M_\ell} = +\infty$.

The latent response is given by

$$I^*_{n\ell} = \lambda_{\ell 0} + \sum_k \lambda_{\ell k} x^*_{nk} + \sigma_{\upsilon\ell}\upsilon_{n\ell}, \qquad \upsilon_{n\ell} \sim \mathcal{N}(0,1).$$

What changes compared to the continuous case? In the ordered probit likelihood (12), probabilities depend only on *standardized differences*

$$\frac{\tau_m - \lambda_{\ell 0} - \sum_k \lambda_{\ell k} x^*_{nk}}{\sigma_{\upsilon\ell}}.$$

As a consequence, the ordered probit layer introduces *two additional invariances* that must be addressed explicitly:

- a *location invariance*: adding the same constant to $I^*_{n\ell}$ and to all thresholds leaves probabilities unchanged;
- a *scale invariance*: multiplying $I^*_{n\ell}$, all thresholds, and $\sigma_{\upsilon\ell}$ by the same positive constant leaves probabilities unchanged.

These invariances are specific to ordinal models and are *in addition* to the invariances already present in the latent-variable structure discussed in the continuous case.

We adopt the same reference-indicator philosophy as in the continuous case. For each latent variable k, one ordinal indicator $\ell(k)$ is chosen as its reference indicator.

- Step 1: anchor the latent variable. Exactly as in the continuous case, we fix

  $$\boxed{\lambda_{\ell(k)0} = 0, \qquad \lambda_{\ell(k)k} = 1 \quad (\text{or } -1).}$$

  This fixes the *location* and *orientation* of the latent variable x^*_{nk} and defines its unit through the response of the reference indicator. This step is conceptually identical to the continuous case and should always be applied first.

- Step 2: fix the location of the threshold system. Because only differences $\tau_m - I^*_{n\ell}$ matter, the threshold system has an arbitrary origin. This must be fixed *once*, and only once.

  There are two principled ways to do so. For non-symmetric thresholds, fix one threshold to zero, for example $\tau_c = 0$. This is a pure location normalization. For symmetric thresholds, impose symmetry around zero. In this case, location is fixed by construction.

  When the Likert scale is designed to be symmetric (which is typically the case), the symmetric parameterization is recommended, because it aligns the statistical model with the semantics of the survey scale and reduces the number of free parameters.

  Note that, in the symmetric case, it is advised to re-parametrize the model. Let M_ℓ be the number of ordered categories. There are $M_\ell - 1$ finite thresholds. When M_ℓ is odd (e.g., 5-point Likert), there is a natural central category. For $M_\ell = 5$, there are four thresholds. A symmetric parameterization is:

$$\boxed{\begin{array}{ll} \begin{aligned} \tau_1 &= -(\delta_1+\delta_2), \\ \tau_2 &= -\delta_1, \\ \tau_3 &= \quad \delta_1, \\ \tau_4 &= \quad (\delta_1+\delta_2), \end{aligned} & \delta_1 > 0,\ \delta_2 > 0. \end{array}}$$

This construction:

- enforces strict ordering automatically;
- fixes the location (centered at zero);
- reduces dimensionality (two parameters instead of four).

When M_ℓ is even, there is no central category. In this case, $M_\ell - 1$ is odd, and symmetry necessarily implies that one threshold lies exactly at zero. For $M_\ell = 4$:

$$\boxed{\tau_1 = -\delta_1, \qquad \tau_2 = 0, \qquad \tau_3 = \delta_1, \qquad \delta_1 > 0.}$$

Here again, location is fixed by symmetry, and ordering is guaranteed.

- Step 3: fix the scale of the ordered probit layer. In ordered probit models, scale is not identified because

$$(\tau_m,\ I^*_{n\ell},\ \sigma_{\upsilon\ell}) \mapsto (a\tau_m,\ aI^*_{n\ell},\ a\sigma_{\upsilon\ell})$$

leaves probabilities unchanged. Therefore, one scale normalization is required *within the ordinal layer*.

To remain consistent with the reference-indicator philosophy adopted for latent variables, we fix the scale of the latent variable by $\lambda_{\ell(k)k} = 1$; and the scale of the ordinal response by fixing $\sigma_{\upsilon\ell} = 1$ for ordinal indicators.

This choice has three advantages:

1. it follows the standard ordered probit convention;
2. it avoids entangling threshold spacings with noise variance;
3. it preserves a clear interpretation of thresholds as cutpoints on a standardized latent response scale.

In summary, for an ordinal indicator ℓ measuring latent variable k, the recommended normalization strategy is:

1. choose $\ell(k)$ as reference indicator;

2. fix $\lambda_{\ell(k)0} = 0$ and $\lambda_{\ell(k)k} = \pm 1$;

3. impose an ordered threshold parameterization;

4. fix threshold location (preferably via symmetry);

5. fix $\sigma_{\upsilon\ell} = 1$.

These steps jointly remove all location and scale indeterminacies, without redundancy, and yield a model whose parameters have a clear, stable, and interpretable meaning.

2 Hybrid choice models

This section builds on the notation and model components introduced in Section 1. We combine the structural equations for the latent variables, the measurement equations for the indicators (continuous or ordinal), and a discrete choice model, into two frameworks of increasing scope: the *MIMIC* model and the *hybrid choice model*.

We use the same notations as in Section 1: n indexes individuals, x_n denotes the observed explanatory variables, x^*_n the vector of latent variables, I_n the vector of observed indicators, and $i_n \in \mathcal{C}_n$ the observed choice.

2.1 The MIMIC model

A *Multiple Indicators Multiple Causes* (MIMIC) model is obtained by combining:

- the structural part ("multiple causes"), which explains each latent variable as a function of observed covariates through the structural equations (2); and

- the measurement part ("multiple indicators"), which explains each indicator as a function of the latent variables through measurement equations (continuous indicators: (4); ordinal indicators: ordered probit, (12)).

The purpose of the MIMIC model is to infer latent variables from their observable manifestations (the indicators) while simultaneously explaining how these latent constructs vary with observed covariates.

For a given individual n, the contribution of the indicators to the likelihood *conditional on the latent variables* x_n^* is obtained by multiplying the indicator-specific contributions:

$$L_n^I(I_n \mid x_n^*, x_n) \;=\; \prod_{\ell=1}^{L_n} \begin{cases} \text{density of Eq. (5),} & \text{if indicator } \ell \text{ is continuous,} \\ \text{probability of Eq. (12),} & \text{if indicator } \ell \text{ is ordinal.} \end{cases} \tag{15}$$

Because x_n^* is not observed, the individual likelihood contribution integrates the conditional indicator likelihood over the distribution of the latent variables implied by the structural equations:

$$L_n^{\text{MIMIC}} \;=\; \int_{x_n^*} L_n^I(I_n \mid x_n^*, x_n)\; f(x_n^* \mid x_n)\; dx_n^*, \tag{16}$$

where $f(x_n^* \mid x_n)$ is the conditional density implied by Eq. (2).

2.2 Hybrid choice model (choice model with latent variables)

A *hybrid choice model* extends the MIMIC model by adding a discrete choice component in which latent variables enter the systematic utilities. The key modeling idea is that attitudes or perceptions (latent variables) may influence choices, while being measured only indirectly through the indicators.

Let $V_{in}(x_n, x_n^*; \beta)$ denote the systematic utility of alternative i for individual n. The probability of the observed choice i_n conditional on the latent variables is

$$P(i_n \mid x_n^*, x_n; \beta) \;=\; \frac{\exp(\mu\, V_{in}(x_n, x_n^*; \beta))}{\sum_{j \in \mathcal{C}_n} \exp(\mu\, V_{jn}(x_n, x_n^*; \beta))}, \tag{17}$$

where μ is the scale parameter of the logit model and β is the vector of utility parameters. Note that the logit model is adopted here for expositional convenience and because it is a widely used specification. However, the framework is fully general and can accommodate alternative discrete choice models, such as nested logit or cross-nested logit models, without any conceptual modification.

For a given individual n, the full observation is (i_n, I_n). Conditional on x_n^*, the hybrid choice model contribution to the likelihood is the product of:

- the conditional choice probability from Eq. (17),
- the conditional indicator likelihood from Eq. (15).

That is,

$$L_n(i_n, I_n \mid x_n^*, x_n) \;=\; P(i_n \mid x_n^*, x_n; \beta)\, L_n^I(I_n \mid x_n^*, x_n)\,. \tag{18}$$

Because x_n^* is latent, the individual likelihood contribution integrates (18) over $f(x_n^* \mid x_n)$ implied by the structural equations (2):

$$L_n^{\text{HCM}} \;=\; \int_{x_n^*} P(i_n \mid x_n^*, x_n; \beta)\, L_n^I(I_n \mid x_n^*, x_n)\;\, f(x_n^* \mid x_n)\, dx_n^*. \tag{19}$$

Let θ denote the full parameter vector, including the parameters of the choice model, structural equations, and measurement equations (and thresholds for ordinal indicators). The sample log-likelihood is

$$\begin{aligned}\mathcal{L}(\theta) &= \sum_n \ln L_n^{\text{HCM}} \\ &= \sum_n \ln \left[\int_{x_n^*} P(i_n \mid x_n^*, x_n; \beta)\, L_n^I(I_n \mid x_n^*, x_n)\;\, f(x_n^* \mid x_n)\, dx_n^* \right].\end{aligned} \tag{20}$$

The integral in (20) typically has no closed form and is evaluated numerically, most often by Monte-Carlo integration.

3 A case study

This example focuses on the estimation of a mode choice model for residents of Switzerland, using revealed preference data. The data were collected as part of a research project aimed to assess the market potential of combined mobility solutions — particularly in urban agglomerations — by identifying the factors that influence individuals in their choice of transport mode (Bierlaire et al., 2011).

The survey was conducted between 2009 and 2010 on behalf of CarPostal, the public transport operator of the Swiss Postal Service. Its primary objective was to collect data on travel behavior in low-density areas, which represent the typical service environment of CarPostal. In addition to revealed preference data, the survey includes several psychometric indicators, enabling the incorporation of latent variables into the model specification.

The data file as well as its description is available on the Biogeme webpage. A description of the variables is also available in Appendix 5.

We consider a model involving two latent variables. The first one captures a “car-centric” attitude. The second one captures an “environmental attitude”. The car-centric attitude captures the extent to which individuals exhibit a strong preference for private car use as their primary mode of transportation. This latent construct reflects values such as independence, flexibility, comfort, and perceived status associated with driving. Individuals with a high car-centric attitude are more likely to perceive cars as the most practical and desirable means of travel, often resisting modal shift to public transport or active mobility. The environmental attitude represents the degree to which individuals value environmental protection and sustainability in their mobility choices. It reflects concerns about issues such as climate change, air pollution, energy consumption, and the broader environmental impacts of transportation. Individuals with a strong environmental attitude are more likely to favor low-emission travel options, to support policies that reduce car use, and to accept constraints on private mobility when these contribute to environmental goals.

3.1 Psychometric indicators

The psychmometric indicators selected to be used in the model are the following:

Envir01 Fuel price should be increased to reduce congestion and air pollution.

Envir02 More public transportation is needed, even if taxes are set to pay the additional costs.

Envir03 Ecology disadvantages minorities and small businesses.

Envir04 People and employment are more important than the environment.

Envir05 I am concerned about global warming.

Envir06 Actions and decision making are needed to limit greenhouse gas emissions.

Mobil03 I use the time of my trip in a productive way.

Mobil05 I reconsider frequently my mode choice.

Mobil08 I do not feel comfortable when I travel close to people I do not know.

Mobil09 Taking the bus helps making the city more comfortable and welcoming.

Mobil10 It is difficult to take the public transport when I travel with my children.

Mobil12 It is very important to have a beautiful car.

LifSty01 I always choose the best products regardless of price.

LifSty07 The pleasure of having something beautiful consists in showing it.

NbCar Number of cars in the household.

The specification of the measurement model relies on two sets of indicators, one for each latent variable. The *car-centric* attitude is measured using the indicators `Envir01`, `Envir02`, `Envir06`, `Mobil03`, `Mobil05`, `Mobil08`, `Mobil09`, `Mobil10`, `LifSty07`, and `NbCar`. These indicators capture aspects related to the perceived convenience, comfort, flexibility, and social meaning of private car use, as well as practical constraints associated with alternative modes. The *environmental* attitude is measured using the indicators `Envir01`, `Envir02`, `Envir03`, `Envir04`, `Envir05`, `Envir06`, `Mobil12`, `LifSty01`, and `NbCar`, which reflect concerns about environmental protection, sustainability, and the trade-offs between environmental objectives, economic considerations, and personal consumption preferences.

The composition of these indicator sets is not imposed *a priori* but results from a combination of theoretical considerations and an iterative modeling process. Indicators were selected based on their conceptual relevance to each latent construct and on empirical performance during estimation.

The indicator `NbCar` differs in nature from the other indicators used in the measurement model. It is not a psychometric indicator based on attitudinal statements evaluated on a Likert scale, but an observed household characteristic reporting the number of cars owned by the household. This variable can take the discrete values 0, 1, 2, and 3. Although `NbCar` does not directly measure attitudes or perceptions, it provides valuable information about underlying latent constructs related to mobility preferences and environmental values. In particular, car ownership reflects long-term mobility decisions and constraints that are strongly associated with both car-centric and environmental attitudes. For this reason, `NbCar` is incorporated into the model as an indicator.

3.2 Structural equations

For the structural equations, we use the linear specification in Eq. (2). The set of explanatory variables included in each structural equation follows the specification used in the implementation. In particular, the car-centric latent variable $x^*_{n,\text{car}}$ is specified as a function of `high_education`, `top_manager`, `employees`, `age_30_less`, `ScaledIncome`, and `car_oriented_parents`. The environmental latent variable $x^*_{n,\text{envir}}$ is specified as a function of `childSuburb`, `ScaledIncome`, `city_center_as_kid`, `artisans`, `high_education`, and `low_education`. These variables are constructed from the raw survey information during data preparation (e.g., `ScaledIncome` is computed as `CalculatedIncome`/1000, `age_30_less` is the indicator `age` $\leqslant 30$, `childSuburb` identifies individuals who lived in suburban areas as children, and `car_oriented_parents` identifies respondents reporting very frequent car use by parents). The final specification of the structural equations results from a combination of behavioral assumptions and empirical trial-and-error, balancing interpretability, parameter stability, and overall fit.

3.3 Measurement equations

For each individual n and each indicator ℓ described in Section 3.1, we introduce a latent continuous response variable, as outlined in Section 1.3. This latent response captures the unobserved propensity underlying the observed ordinal response on a Likert scale.

For the indicators associated with the car-centric attitude, the latent response is modeled as:

$$I^*_{n\ell} = \lambda_{0\ell} + \lambda_{1\ell} x^*_{n,\text{car}} + \lambda_{2\ell} \upsilon_{n\ell}, \tag{21}$$

where $\lambda_{0\ell}$ is an intercept term, $\lambda_{1\ell}$ is the loading on the latent variable $x^*_{n,\text{car}}$, $\lambda_{2\ell}$ scales the stochastic component, and $\upsilon_{n\ell}$ is a random error term.

The indicator `Envir01` is selected for the normalization of the measurement model. Individuals with a stronger car-centric attitude are expected to be more likely to *disagree* with the corresponding statement. Accordingly, the loading $\lambda_{1\ell}$ is expected to be negative, and fixed to -1 to establish the direction of the latent construct. The scale parameter $\lambda_{2\ell}$ is normalized to 1 to ensure identifiability of the model.

Similarly, for the indicators capturing the environmental attitude, we specify:

$$I^*_{n\ell} = \lambda_{0\ell} + \lambda_{1\ell} x^*_{n,\text{env}} + \lambda_{2\ell} \upsilon_{n\ell}, \tag{22}$$

with analogous interpretation of the parameters.

The indicator `Envir02` is selected for the normalization of this measurement model. Individuals with a stronger urban-preference attitude are expected to be more likely to *agree* with the corresponding statement. Accordingly, the loading $\lambda_{1\ell}$ is expected to be positive, and fixed to 1 to establish the direction of the latent construct. The scale parameter $\lambda_{2\ell}$ is normalized to 1 to ensure identifiability of the model.

The threshold specification follows directly from the normalization and identification principles discussed in Section 1.4 and from the ordered probit formulation introduced in Section 1.3. In particular, threshold parameterizations are chosen so as to (i) enforce the ordering constraints, (ii) fix the location of the ordinal response scale exactly once, and (iii) remain consistent with the reference-indicator strategy adopted for the latent variables.

For Likert-type indicators with five response categories, we use a symmetric threshold parameterization centered at zero:

$$\begin{aligned}\tau_1 &= -(\delta_1 + \delta_2),\\ \tau_2 &= -\delta_1,\\ \tau_3 &= \delta_1,\\ \tau_4 &= (\delta_1 + \delta_2),\end{aligned}$$

where $\delta_1 > 0$ and $\delta_2 > 0$ are estimated. This parameterization has three desirable properties. First, it guarantees strict ordering of the thresholds by construction. Second, it fixes the location of the threshold system by centering it at zero, thereby removing the location indeterminacy inherent to ordered probit models. Third, it reflects the semantic symmetry of standard Likert scales (e.g., from "strongly disagree" to "strongly agree") while reducing the number of free parameters. These thresholds are shared across all Likert-type indicators, reflecting the modeling assumption that the response categories have a comparable interpretation across statements.

The indicator `NbCar` is treated separately, as it is not a psychometric Likert-scale indicator but an observed household characteristic reporting the number of cars owned. Although it is used as an indicator of the latent constructs, its response scale is inherently asymmetric and quantitative. Since `NbCar` takes four ordered values, three thresholds are required. For this indicator, we adopt a non-symmetric threshold parameterization and fix the first threshold to zero:

$$\begin{aligned}\tau_1^{\text{NbCar}} &= 0,\\ \tau_2^{\text{NbCar}} &= \tau_1^{\text{NbCar}} + \delta_1^{\text{NbCar}},\\ \tau_3^{\text{NbCar}} &= \tau_2^{\text{NbCar}} + \delta_2^{\text{NbCar}},\end{aligned}$$

where $\delta_1^{\text{NbCar}} > 0$ and $\delta_2^{\text{NbCar}} > 0$ are estimated. Fixing $\tau_1^{\text{NbCar}} = 0$ provides the required location normalization for this indicator, while the positive incremental parameterization ensures ordered thresholds without imposing symmetry. This choice is fully consistent with the overall normalization strategy: location is fixed once for the ordinal layer, and the scale of the latent variables remains anchored through the reference indicators.

3.4 Implementation notes

The results reported below are produced using the set of Python specification files included in the appendix (Sections 6.1–6.14). These files have been developed and tested with `Biogeme 3.3.2`. As Biogeme evolves, minor adaptations of the syntax may be required in future versions. The goal of the implementation is to keep the various model variants (choice-only, MIMIC, and hybrid choice; maximum likelihood and Bayesian estimation) consistent by relying on shared specification components and a centralized configuration mechanism. This subsection summarizes the role of each file and the type of specification information it contains.

Data preparation (6.1). The file in Section 6.1 is a standard Biogeme data preparation script. It reads the raw data, applies the sample cleaning rules (e.g., removal of inconsistent observations), and constructs the derived variables used throughout the model specification (such as scaled income, socio-demographic indicators, and other transformed covariates). The resulting `Database` object constitutes the common input for all estimation variants.

Indicator definitions and threshold conventions (6.4). The file in Section 6.4 provides the complete list of indicators used in the measurement model. For each indicator, it stores its identifier (name), the corresponding survey statement, and its type. In the present example, two indicator types are used: `likert` for psychometric indicators collected on a Likert scale, and `cars` for the discrete indicator `NbCar` (number of cars in the household). In addition, the file defines the list of indicator types and the associated threshold conventions. For each type, it specifies whether the thresholds are symmetric or not, the list of admissible response categories, and the "neutral" labels (if any) that are ignored in estimation. These definitions ensure that all measurement equations and threshold specifications are generated consistently across the model variants.

Latent variable definitions (6.3). The file in Section 6.3 defines the latent variables used in the case study and, for each of them, the list of explanatory variables entering its structural equation. This file therefore contains the substantive specification choices for the structural part of the model: the names of the latent constructs and the observed covariates assumed to explain them.

Central configuration (6.2). To ensure that all model variants are generated from the same building blocks, the implementation relies on a configuration object defined in Section 6.2. This configuration determines which components are active and how estimation is performed. It contains the following entries:

- `name`: a string identifier used to label the model run and its output files;
- `latent_variables` (`"zero"` or `"two"`): whether the specification includes no latent variables or the two latent variables of the case study;
- `choice_model` (`"yes"` or `"no"`): whether the discrete choice component is included (hybrid/choice-only) or omitted (pure MIMIC);
- `estimation` (`"ml"` or `"bayes"`): whether the model is estimated by maximum likelihood or using Bayesian inference;
- `number_of_bayesian_draws_per_chain`: the number of posterior draws generated per MCMC chain (relevant when `estimation="bayes"`);
- `number_of_monte_carlo_draws`: the number of Monte-Carlo draws used to approximate the integrals over the latent variables (relevant `estimation="ml"`).

This design avoids duplicating code across variants and makes the comparison between specifications transparent.

MIMIC component and normalization (6.5). The file in Section 6.5 builds the MIMIC part of the model (structural and measurement equations) as a function of the configuration. It is also where the reference-indicator normalization is declared explicitly. In particular, the reference indicator used to anchor a latent variable is identified through a normalization object of the form `Normalization(indicator='Envir01', coefficient=-1)`, where the `coefficient` specifies the fixed loading (e.g., -1) used for identification in the corresponding measurement equation. This file therefore centralizes the measurement-structure assumptions and the identification choices of the latent-variable system.

Choice model component (6.6). The file in Section 6.6 contains the specification of the discrete choice model. Depending on the configuration, the choice model is defined either without latent variables (choice-only baseline) or with latent variables entering the utilities (hybrid choice model).

Estimation control and caching of results (6.8). The file in Section 6.8 orchestrates the execution of the estimation in the requested mode (maximum likelihood or Bayesian), based on the configuration. For reproducibility and efficiency, it first checks whether estimation outputs already exist; if so, results are read from disk rather than recomputed. Otherwise, the file triggers a new estimation run. This file therefore handles the "run-or-read" logic that supports systematic experimentation with multiple model variants.

Log-likelihood assembly and estimation (6.7). Finally, the file in Section 6.7 assembles the full model implied by the configuration and generates the corresponding (log-)likelihood expression. This includes combining the relevant components (choice probability, measurement likelihood, structural density) and performing the required integration over latent variables using Monte-Carlo simulation when appropriate. The same file then calls the estimation routines corresponding to the selected estimation paradigm (maximum likelihood or Bayesian). In short, it is the entry point where the complete hybrid choice model likelihood is constructed and estimated.

Estimation scripts (6.9–6.14). These are six driver scripts, each corresponding to one combination of model scope (choice-only, MIMIC, or full hybrid choice) and estimation method (maximum likelihood or Bayesian). Each script defines the appropriate configuration (Section 6.2) and then relies on the generic estimation workflow (Sections 6.8 and 6.7) to either run the estimation or read existing outputs.

- **Choice-only, maximum likelihood** (Section 6.9): the script plot_b01_choice_only_ml.py estimates the discrete choice model without latent variables (latent_variables ="zero", choice_model="yes", estimation="ml"). It provides a baseline choice specification used for comparison with latent-variable extensions.

- **MIMIC, maximum likelihood** (Section 6.10): the script plot_b02_mimic_ml.py estimates the latent-variable system (structural and measurement equations) without the choice component (latent_variables ="two", choice_model="no", estimation="ml"). It focuses on how the latent constructs are explained by covariates and reflected in indicators.

- **Hybrid choice, maximum likelihood** (Section 6.11): the script plot_b03_hybrid_ml.py estimates the full hybrid choice model, combining the choice model with the latent-variable system (latent_variables ="two", choice_model="yes", estimation="ml"). The likelihood integrates the choice and measurement components over the latent variables.

- **Choice-only, Bayesian** (Section 6.12): the script plot_b04_choice_only_bayes.py estimates the discrete choice model without latent variables using Bayesian inference (latent_variables ="zero", choice_model="yes", estimation="bayes"). It provides the Bayesian counterpart of the maximum-likelihood baseline.

- **MIMIC, Bayesian** (Section 6.13): the script plot_b05_mimic_bayes.py estimates the latent-variable system without the choice component using Bayesian inference (latent_variables ="two", choice_model="no", estimation="bayes"). It delivers posterior inference for the structural and measurement parameters.

- **Hybrid choice, Bayesian** (Section 6.14): the script plot_b06_hybrid_bayes.py estimates the complete hybrid choice model using Bayesian inference (latent_variables ="two", choice_model="yes", estimation="bayes"). It is the most comprehensive variant and yields posterior distributions for both the choice parameters and the latent-variable system, accounting for the full joint likelihood.

All six scripts rely on the same underlying model specification files (Sections 6.3, 6.4, 6.5, and 6.6); the differences between them arise solely from the configuration settings and the chosen estimation paradigm.

4 Conclusion

Choice models with latent variables offer a powerful and flexible framework for capturing complex behavioral mechanisms underlying decision-making. By incorporating unobserved psychological constructs such as attitudes, and perceptions, these models extend the explanatory power of traditional discrete choice models. They allow researchers to account for systematic heterogeneity in behavior that is not directly observed in the data, thereby enhancing both the behavioral realism and predictive performance of the models.

Despite their potential, these models are inherently more complex to specify, estimate, and interpret. It is therefore recommended to proceed incrementally. A practical and effective strategy is to begin by developing and estimating the choice model and the MIMIC model independently. This allows the analyst to ensure that both components are correctly specified and empirically supported.

Once the separate models have been validated, the next step is to explore their integration through sequential estimation. In this stage, the latent variables generated from the MIMIC model are incorporated into the utility specification of the choice model. This provides valuable insights into how these latent constructs influence behavior, while still maintaining manageable computational complexity.

Only after the specification has been refined and the results from the sequential estimation are deemed satisfactory should one proceed to the simultaneous estimation of all components. This final step — though computationally more demanding — offers the benefit of statistical efficiency by leveraging all available information jointly. It also provides a more coherent treatment of the latent variables, since their estimation is informed not only by the indicators, but also by the observed choices.

5 Description of the variables

The following table describes the variables involved in the models described in this document.

Name	Description
TimePT	The duration of the loop performed in public transport (in minutes).
WaitingTimePT	The total waiting time in a loop performed in public transports (in minutes).
TimeCar	The total duration of a loop made using the car (in minutes).
MarginalCostPT	The total cost of a loop performed in public transports, taking into account the ownership of a seasonal ticket by the respondent. If the respondent has a "GA" (full Swiss season ticket), a seasonal ticket for the line or the area, this variable takes value zero. If the respondent has a half-fare travelcard, this variable corresponds to half the cost of the trip by public transport..
CostCarCHF	The total gas cost of a loop performed with the car in CHF.
TripPurpose	The main purpose of the loop: 1 =Work-related trips; 2 =Work- and leisure-related trips; 3 =Leisure related trips. -1 represents missing values
UrbRur	Binary variable, where: 1 =Rural; 2 =Urban.
distance_km	Total distance performed for the loop.
age	Age of the respondent (in years) -1 represents missing values.

ResidChild	Main place of residence as a kid (< 18), 1 is city center (large town), 2 is city center (small town), 3 is suburbs, 4 is suburban town, 5 is country side (village), 6 is countryside (isolated), -1 is for missing data and -2 if respondent didn't answer to any opinion questions.
NbCar	Number of cars in the household.-1 for missing value.
NbBicy	Number of bikes in the household. -1 for missing value.
HouseType	House type, 1 is individual house (or terraced house), 2 is apartment (and other types of multi-family residential), 3 is independent room (subletting). -1 for missing value.
Income	Net monthly income of the household in CHF. 1 is less than 2500, 2 is from 2501 to 4000, 3 is from 4001 to 6000, 4 is from 6001 to 8000, 5 is from 8001 to 10'000 and 6 is more than 10'001. -1 for missing value.
CalculatedIncome	Net monthly income of the household in CHF, calculated as a continuous variable. The value is the center of the interval of the corresponding incone category.
FamilSitu	Familiar situation: 1 is single, 2 is in a couple without children, 3 is in a couple with children, 4 is single with your own children, 5 is in a colocation, 6 is with your parents and 7 is for other situations. -1 for missing values.
SocioProfCat	To which of the following socioprofessional categories do you belong? 1 is for top managers, 2 for intellectual professions, 3 for freelancers, 4 for intermediate professions, 5 for artisans and salespersons, 6 for employees, 7 for workers and 8 for others. -1 for missing values.
GenAbST	Is equal to 1 if the respondent has a GA (full Swiss season ticket) and 2 if not.

Education	Highest education achieved. As mentioned by Wikipedia in English: "The education system in Switzerland is very diverse, because the constitution of Switzerland delegates the authority for the school system mainly to the cantons. The Swiss constitution sets the foundations, namely that primary school is obligatory for every child and is free in public schools and that the confederation can run or support universities." (source: Education in Switzerland (Wikipedia), accessed April 16, 2013). It is thus difficult to translate the survey that was originally in French and German. The possible answers in the survey are: 1. Unfinished compulsory education: education is compulsory in Switzerland but pupils may finish it at the legal age without succeeding the final exam. 2. Compulsory education with diploma. 3. Vocational education: a three or four-year period of training both in a company and following theoretical courses. Ends with a diploma called "Certificat fédéral de capacité" (i.e., "professional baccalaureate") (reference: Certificat fédéral de capacité (Wikipedia) - in French). 4. A 3-year generalist school giving access to teaching school, nursing schools, social work school, universities of applied sciences or vocational education (sometime in less than the normal number of years). It does not give access to universities in Switzerland. 5. High school: ends with the general baccalaureate exam. The general baccalaureate gives access automatically to universities. 6. Universities of applied sciences, teaching schools, nursing schools, social work schools: ends with a Bachelor and sometimes a Master, mostly focus on vocational training. 7. Universities and institutes of technology: ends with an academic Bachelor and in most cases an academic Master. 8. PhD thesis.

Table 1: Description of variables

6 Complete specification files

This section presents the Python implementation of the hybrid choice model used in the case study. The following specification files have been used for the estimation of the results presented in this chapter. They have been developed and tested with `Biogeme 3.3.2`. It is possible that minor adaptations of the syntax may be required for future versions of Biogeme.

The files are organized by *role* rather than by estimation approach: data preparation and configuration, model specification (latent variables, indicators, MIMIC and choice components), estimation workflow, and result visualization. This structure mirrors the modeling logic developed in the previous sections and allows the same core model to be estimated under different assumptions (choice-only, MIMIC, or full hybrid model; maximum likelihood or Bayesian estimation).

Data preparation and configuration

These files define the dataset, variable transformations, and global configuration options shared across all model variants.

6.1 optima.py

```
"""
.. _optima_data:

Data preparation for Optima
===========================

Prepare data for the Optima case study.

:author: Michel Bierlaire
:date: Wed Apr 12 20:52:37 2023

"""

import pandas as pd

import biogeme.database as db
from biogeme.expressions import Variable

data_file_path = 'optima.dat'


# %%
# Read the data
def read_data() -> db.Database:
    """Read the data from file"""
    df = pd.read_csv(data_file_path, sep='\t')
    # Exclude observations such that the chosen alternative is -1
    df.drop(df[df['Choice'] == -1].index, inplace=True)
    # Exclude non workers
    df = df[df['OccupStat'].isin([1, 2])]
    # Exclude tours with 1 trip
    df.drop(df[df["NbTrajects"] == 1].index, inplace=True)
    # Exclude tours longer than 100km
    # df.drop(df[df["distance_km"] > 100].index, inplace=True)
    # Exclude zero travel time
    df.drop(df[df["TimePT"] == 0].index, inplace=True)
    df.drop(df[df["TimeCar"] == 0].index, inplace=True)
    df.drop(df[df["distance_km"] == 0].index, inplace=True)

    car_not_available = df['CarAvail'] == 3
    car_is_chosen = df['Choice'] == 1
    incompatible = car_is_chosen & car_not_available
    df.drop(df[incompatible].index, inplace=True)

    df['worker'] = df['OccupStat'].isin([1, 2]).astype(int)
    df['car_is_available'] = df['CarAvail'] != 3
    # Normalize the weights
    sum_weight = df['Weight'].sum()
    number_of_rows = df.shape[0]
    df['normalized_weight'] = df['Weight'] * number_of_rows / sum_weight
    # Group car ownership: 3, 4, and 6 cars are grouped as 3
    df['number_of_cars'] = df['NbCar'].replace({3: 3, 4: 3, 6: 3})
    database = db.Database(name=data_file_path, dataframe=df)
    _ = database.define_variable('livesInUrbanArea', UrbRur == 2)
    _ = database.define_variable('owningHouse', OwnHouse == 1)
    _ = database.define_variable('used_to_go_to_school_by_car', ModeToSchool == 1)
    _ = database.define_variable('city_center_as_kid', ResidChild == 1)
    _ = database.define_variable('ScaledIncome', CalculatedIncome / 1000)
    _ = database.define_variable('age_65_more', age >= 65)
    _ = database.define_variable('age_30_less', age <= 30)
    _ = database.define_variable('age_category', 2 - (age <= 30) + (age >= 65))
    _ = database.define_variable(
        'household_size', 3 - 2 * (NbHousehold == 1) - (NbHousehold == 2)
    )
    # 15% / 50% / 85 % quantiles of the income distribution in the data
    _ = database.define_variable(
        'income_category',
        1
        + (CalculatedIncome >= 3250)
        + (CalculatedIncome >= 7000)
        + (CalculatedIncome >= 15000),
    )
    # % 30% / 60% quantile of the distance in the data
    _ = database.define_variable('distance_category', 1 + (distance_km >= 50))

    _ = database.define_variable('moreThanOneCar', NbCar > 1)
    _ = database.define_variable('moreThanOneBike', NbBicy > 1)
    _ = database.define_variable('individualHouse', HouseType == 1)
    _ = database.define_variable('male', Gender == 1)
    _ = database.define_variable('single', ((FamilSitu == 1) + (FamilSitu == 4)) > 0)
    _ = database.define_variable(
        'haveChildren', ((FamilSitu == 3) + (FamilSitu == 4)) > 0
    )
    _ = database.define_variable('haveGA', GenAbST == 1)
    _ = database.define_variable('high_education', Education >= 6)
    _ = database.define_variable('low_education', Education <= 3)
    _ = database.define_variable('top_manager', SocioProfCat == 1)
    _ = database.define_variable('artisans', SocioProfCat == 5)
    _ = database.define_variable('employees', SocioProfCat == 6)
    _ = database.define_variable(
        'childCenter', ((ResidChild == 1) + (ResidChild == 2)) > 0
    )

    _ = database.define_variable(
        'childSuburb', ((ResidChild == 3) + (ResidChild == 4)) > 0
    )
    _ = database.define_variable('car_oriented_parents', FreqCarPar == 4)
    _ = database.define_variable('TimePT_scaled', TimePT / 200)
    _ = database.define_variable('TimePT_hour', TimePT / 60)
    _ = database.define_variable('TimeCar_scaled', TimeCar / 200)
    _ = database.define_variable('TimeCar_hour', TimeCar / 60)
    _ = database.define_variable('MarginalCostPT_scaled', MarginalCostPT / 10)
    _ = database.define_variable('CostCarCHF_scaled', CostCarCHF / 10)
    _ = database.define_variable('distance_km_scaled', distance_km / 5)
    _ = database.define_variable('PurpHWH', TripPurpose == 1)
    _ = database.define_variable('PurpOther', TripPurpose != 1)
    _ = database.define_variable(
        'number_of_trips',
        (NbTrajects == 1) + 2 * (NbTrajects == 2) + 3 * (NbTrajects >= 3),
    )
    # urbanization: 1 = urban, 2 = mixed, 3 = rural
    _ = database.define_variable(
        'urbanization', 2 - 1 * (TypeCommune <= 3) + 1 * (TypeCommune >= 8)
    )

    return database


# %%
# Variables from the data
Choice = Variable('Choice')
TimePT = Variable('TimePT')
```

```
TimeCar = Variable(’TimeCar’)
MarginalCostPT = Variable(’MarginalCostPT’)
CostCarCHF = Variable(’CostCarCHF’)
distance_km = Variable(’distance_km’)
Gender = Variable(’Gender’)
OccupStat = Variable(’OccupStat’)
Weight = Variable(’Weight’)
ID = Variable(’ID’)
DestAct = Variable(’DestAct’)
NbTransf = Variable(’NbTransf’)
WalkingTimePT = Variable(’WalkingTimePT’)
WaitingTimePT = Variable(’WaitingTimePT’)
CostPT = Variable(’CostPT’)
CostCar = Variable(’CostCar’)
NbHousehold = Variable(’NbHousehold’)
NbChild = Variable(’NbChild’)
NbCar = Variable(’NbCar’)
number_of_cars = Variable(’number_of_cars’)
NbMoto = Variable(’NbMoto’)
NbBicy = Variable(’NbBicy’)
NbBicyChild = Variable(’NbBicyChild’)
NbComp = Variable(’NbComp’)
NbTV = Variable(’NbTV’)
Internet = Variable(’Internet’)
NewsPaperSubs = Variable(’NewsPaperSubs’)
NbCellPhones = Variable(’NbCellPhones’)
NbSmartPhone = Variable(’NbSmartPhone’)
HouseType = Variable(’HouseType’)
OwnHouse = Variable(’OwnHouse’)
NbRoomsHouse = Variable(’NbRoomsHouse’)
YearsInHouse = Variable(’YearsInHouse’)
Income = Variable(’Income’)
BirthYear = Variable(’BirthYear’)
Mothertongue = Variable(’Mothertongue’)
FamilSitu = Variable(’FamilSitu’)
SocioProfCat = Variable(’SocioProfCat’)
CalculatedIncome = Variable(’CalculatedIncome’)
Education = Variable(’Education’)
HalfFareST = Variable(’HalfFareST’)
LineRelST = Variable(’LineRelST’)
GenAbST = Variable(’GenAbST’)
AreaRelST = Variable(’AreaRelST’)
OtherST = Variable(’OtherST’)
urbanization = Variable(’urbanization’)
three_trips_or_more = Variable(’three_trips_or_more’)
CarAvail = Variable(’CarAvail’)
Envir01 = Variable(’Envir01’)
Envir02 = Variable(’Envir02’)
Envir03 = Variable(’Envir03’)
Envir04 = Variable(’Envir04’)
Envir05 = Variable(’Envir05’)
Envir06 = Variable(’Envir06’)
Mobil01 = Variable(’Mobil01’)
Mobil02 = Variable(’Mobil02’)
Mobil03 = Variable(’Mobil03’)
Mobil04 = Variable(’Mobil04’)
Mobil05 = Variable(’Mobil05’)
Mobil06 = Variable(’Mobil06’)
Mobil07 = Variable(’Mobil07’)
Mobil08 = Variable(’Mobil08’)
Mobil09 = Variable(’Mobil09’)
Mobil10 = Variable(’Mobil10’)
Mobil11 = Variable(’Mobil11’)
Mobil12 = Variable(’Mobil12’)
Mobil13 = Variable(’Mobil13’)
Mobil14 = Variable(’Mobil14’)
Mobil15 = Variable(’Mobil15’)
Mobil16 = Variable(’Mobil16’)
Mobil17 = Variable(’Mobil17’)
Mobil18 = Variable(’Mobil18’)
Mobil19 = Variable(’Mobil19’)
Mobil20 = Variable(’Mobil20’)
Mobil21 = Variable(’Mobil21’)
Mobil22 = Variable(’Mobil22’)
Mobil23 = Variable(’Mobil23’)
Mobil24 = Variable(’Mobil24’)
Mobil25 = Variable(’Mobil25’)
Mobil26 = Variable(’Mobil26’)
Mobil27 = Variable(’Mobil27’)
ResidCh01 = Variable(’ResidCh01’)
ResidCh02 = Variable(’ResidCh02’)
ResidCh03 = Variable(’ResidCh03’)
ResidCh04 = Variable(’ResidCh04’)
```

```
ResidCh05 = Variable('ResidCh05')
ResidCh06 = Variable('ResidCh06')
ResidCh07 = Variable('ResidCh07')
LifSty01 = Variable('LifSty01')
LifSty02 = Variable('LifSty02')
LifSty03 = Variable('LifSty03')
LifSty04 = Variable('LifSty04')
LifSty05 = Variable('LifSty05')
LifSty06 = Variable('LifSty06')
LifSty07 = Variable('LifSty07')
LifSty08 = Variable('LifSty08')
LifSty09 = Variable('LifSty09')
LifSty10 = Variable('LifSty10')
LifSty11 = Variable('LifSty11')
LifSty12 = Variable('LifSty12')
LifSty13 = Variable('LifSty13')
LifSty14 = Variable('LifSty14')
TripPurpose = Variable('TripPurpose')
TypeCommune = Variable('TypeCommune')
UrbRur = Variable('UrbRur')
LangCode = Variable('LangCode')
ClassifCodeLine = Variable('ClassifCodeLine')
frequency = Variable('frequency')
ResidChild = Variable('ResidChild')
NbTrajects = Variable('NbTrajects')
FreqCarPar = Variable('FreqCarPar')
FreqTrainPar = Variable('FreqTrainPar')
FreqOtherPar = Variable('FreqOtherPar')
FreqTripHouseh = Variable('FreqTripHouseh')
Region = Variable('Region')
InVehicleTime = Variable('InVehicleTime')
ModeToSchool = Variable('ModeToSchool')
ReportedDuration = Variable('ReportedDuration')
CoderegionCAR = Variable('CoderegionCAR')
age = Variable('age')
age_category = Variable('age_category')
normalized_weight = Variable('normalized_weight')

ScaledIncome = Variable('ScaledIncome')
age_65_more = Variable('age_65_more')
moreThanOneCar = Variable('moreThanOneCar')
moreThanOneBike = Variable('moreThanOneBike')
individualHouse = Variable('individualHouse')
male = Variable('male')
haveChildren = Variable('haveChildren')
haveGA = Variable('haveGA')
high_education = Variable('high_education')
low_education = Variable('low_education')
childCenter = Variable('childCenter')
childSuburb = Variable('childSuburb')
TimePT_scaled = Variable('TimePT_scaled')
TimePT_hour = Variable('TimePT_hour')
TimeCar_scaled = Variable('TimeCar_scaled')
TimeCar_hour = Variable('TimeCar_hour')
MarginalCostPT_scaled = Variable('MarginalCostPT_scaled')
CostCarCHF_scaled = Variable('CostCarCHF_scaled')
distance_km_scaled = Variable('distance_km_scaled')
PurpHWH = Variable('PurpHWH')
PurpOther = Variable('PurpOther')
livesInUrbanArea = Variable('livesInUrbanArea')
household_size = Variable('household_size')
income_category = Variable('income_category')
distance_category = Variable('distance_category')
worker = Variable('worker')
car_is_available = Variable('car_is_available')
```

6.2 config.py

```
from dataclasses import dataclass
from typing import Literal


@dataclass(frozen=True)
class Config:
    name: str
    latent_variables: Literal["zero", "two"]
    choice_model: Literal["yes", "no"]
    estimation: Literal["bayes", "ml"]
    number_of_bayesian_draws_per_chain: int
    number_of_monte_carlo_draws: int
```

Latent variables and measurement structure

The following files define the latent variables, the associated psychometric and non-psychometric indicators, and the MIMIC component (structural and measurement equations without choices).

6.3 latent_variables .py

```python
# Latent variable for the car centric attitude
car_explanatory_variables: list[str] = [
    'high_education',
    'top_manager',
    'employees',
    'age_30_less',
    'ScaledIncome',
    'car_oriented_parents',
]

car_name = 'car_centric_attitude'
car_likert_indicators: set[str] = {
    'Envir01',
    'Envir02',
    'Envir06',
    'Mobil03',
    'Mobil05',
    'Mobil08',
    'Mobil09',
    'Mobil10',
    'LifSty07',
    'NbCar',
}

# Latent variable for the environmental attitude
environment_explanatory_variables: list[str] = [
    'childSuburb',
    'ScaledIncome',
    'city_center_as_kid',
    'artisans',
    'high_education',
    'low_education',
]

env_name = 'environmental_attitude'
environment_likert_indicators: set[str] = {
    'Envir01',
    'Envir02',
    'Envir03',
    'Envir04',
    'Envir05',
    'Envir06',
    'Mobil12',
    'LifSty01',
    'NbCar',
}
```

6.4 likert_indicators .py

```python
from biogeme.latent_variables import LikertIndicator
from biogeme.latent_variables.likert_indicators import LikertType

likert_indicators = [
    LikertIndicator(
        name='Envir01',
        statement='Fuel price should be increased to reduce congestion and air
            pollution.',
        type='likert',
    ),
    LikertIndicator(
        name='Envir02',
        statement='More public transportation is needed, even if taxes are set to pay
            the additional costs.',
        type='likert',
    ),
    LikertIndicator(
        name='Envir03',
        statement='Ecology disadvantages minorities and small businesses.',
        type='likert',
    ),
    LikertIndicator(
        name='Envir04',
        statement='People and employment are more important than the environment.',
        type='likert',
    ),
    LikertIndicator(
        name='Envir05',
        statement='I am concerned about global warming.',
        type='likert',
    ),
    LikertIndicator(
        name='Envir06',
        statement='Actions and decision making are needed to limit greenhouse gas emissions.',
        type='likert',
    ),
    LikertIndicator(
        name='Mobil03',
        statement='I use the time of my trip in a productive way.',
        type='likert',
    ),
    LikertIndicator(
        name='Mobil05',
        statement='I reconsider frequently my mode choice.',
        type='likert',
    ),
    LikertIndicator(
        name='Mobil08',
        statement='I do not feel comfortable when I travel close to people I do not know.',
        type='likert',
    ),
    LikertIndicator(
        name='Mobil09',
        statement='Taking the bus helps making the city more comfortable and welcoming.',
        type='likert',
    ),
    LikertIndicator(
        name='Mobil10',
        statement='It is difficult to take the public transport when I travel with my children.',
        type='likert',
    ),
    LikertIndicator(
        name='Mobil12',
        statement='It is very important to have a beautiful car.',
        type='likert',
    ),
    LikertIndicator(
        name='LifSty01',
        statement='I always choose the best products regardless of price.',
        type='likert',
    ),
    LikertIndicator(
        name='LifSty07',
        statement='The pleasure of having something beautiful consists in showing it.',
        type='likert',
    ),
    LikertIndicator(
        name='NbCar',
        statement='Number of cars in the household',
        type='cars',
    ),
]

likert_types = [
    LikertType(
        type='likert',
        symmetric=True,
        categories=[1, 2, 3, 4, 5],
        neutral_labels=[6, -1],
    ),
    LikertType(
        type='cars',
        symmetric=False,
        categories=[0, 1, 2, 3],
        neutral_labels=[-1],
        fix_first_cut_point_for_non_symmetric_thresholds=0.0,
    ),
]
```

6.5 mimic.py

```python
from biogeme.latent_variables import (
    EstimationMode,
    LatentVariable,
    Normalization,
    OrderedMimic,
    StructuralEquation,
)

from config import Config
from latent_variables import (
    car_explanatory_variables,
    car_likert_indicators,
    car_name,
    env_name,
    environment_explanatory_variables,
    environment_likert_indicators,
)
from likert_indicators import likert_indicators, likert_types


def generate_mimic_model(config: Config):
    bayesian_estimation = config.estimation == "bayes"
    estimation_mode = (
        EstimationMode.BAYESIAN
        if bayesian_estimation
        else EstimationMode.MAXIMUM_LIKELIHOOD
    )

    mimic_model = OrderedMimic(
        estimation_mode=estimation_mode,
        likert_indicators=likert_indicators,
        likert_types=likert_types,
    )

    car_lv = LatentVariable(
        name=car_name,
        structural_equation=StructuralEquation(
            name=car_name,
            explanatory_variables=car_explanatory_variables,
        ),
        indicators=car_likert_indicators,
        normalization=Normalization(indicator='Envir01', coefficient=-1),
    )
    mimic_model.add_latent_variable(car_lv)

    env_lv = LatentVariable(
        name=env_name,
        structural_equation=StructuralEquation(
            name=env_name,
            explanatory_variables=environment_explanatory_variables,
        ),
        indicators=environment_likert_indicators,
        normalization=Normalization(indicator='Envir02', coefficient=1),
    )
    mimic_model.add_latent_variable(env_lv)
    return mimic_model
```

Choice model specification

This file defines the discrete choice component and its interaction with the latent variables in the hybrid choice model.

6.6 choice_model.py

```python
from biogeme.expressions import Beta, Expression, Numeric, exp

from config import Config
from latent_variables import car_name, env_name
from mimic import generate_mimic_model
from optima import (
    CostCarCHF,
    MarginalCostPT,
    PurpHWH,
    TimeCar_hour,
    TimePT_hour,
    WaitingTimePT,
    distance_km,
)


def generate_choice_model(config: Config) -> dict[int, Expression]:
    """Generate the choice model utilities.

    The behavior depends on the number of latent variables requested:

    - ``config.latent_variables == 'zero'``: no latent variables enter the choice model.
    - ``config.latent_variables == 'one'``: only the car-centric latent variable enters.
    - ``config.latent_variables == 'two'``: both car-centric and environmental latent variables enter.

    Note: this function returns only the utilities. The estimation / likelihood wrapping
    is handled elsewhere.
    """
    include_latent_variables = config.latent_variables == "two"

    # Latent variables can be: zero, one (car-centric only), or two (car-centric + environmental).
    car_centric_attitude = None
    environmental_attitude = None

    if include_latent_variables:
        mimic = generate_mimic_model(config=config)
        car_centric_attitude = mimic.get_latent_variable(name=car_name)
        environmental_attitude = mimic.get_latent_variable(name=env_name)

    # %%
    # Choice model
    work_trip = PurpHWH == 1
    other_trip_purposes = PurpHWH != 1

    # Choice model: parameters
    choice_beta_cost = Beta('choice_beta_cost', 0, None, 0, 0)

    choice_asc_car = Beta('choice_asc_car', 0.0, None, None, 0)

    choice_asc_pt = Beta('choice_asc_pt', 0, None, None, 0)

    choice_beta_dist_work = Beta('choice_beta_dist_work', 0, None, 0, 0)
    choice_beta_dist_other_purposes = Beta(
        'choice_beta_dist_other_purposes', 0, None, 0, 0
    )
    choice_beta_dist = (
        choice_beta_dist_work * work_trip
        + choice_beta_dist_other_purposes * other_trip_purposes
    )

    # Time coefficients with optional LV interactions
    choice_beta_time_car_ref = Beta('choice_beta_time_car_ref', 0, None, 0, 0)
    choice_beta_time_car = choice_beta_time_car_ref

    if include_latent_variables:
        beta_time_car_lambda_environment = Beta(
            'beta_time_car_lambda_environment', -1, None, 0, 0
        )
        choice_beta_time_car *= exp(
            beta_time_car_lambda_environment
            * environmental_attitude.structural_equation_jax
        )

        beta_time_car_lambda_car_centric = Beta(
            'beta_time_car_lambda_car_centric', -1, None, 0, 0
        )
        choice_beta_time_car *= exp(
            beta_time_car_lambda_car_centric
            * car_centric_attitude.structural_equation_jax
        )

    choice_beta_time_pt_ref = Beta('choice_beta_time_pt_ref', 0, None, 0, 0)
    choice_beta_time_pt = choice_beta_time_pt_ref

    if include_latent_variables:
        beta_time_pt_lambda_environment = Beta(
            'beta_time_pt_lambda_environment', -1, None, 0, 0
        )
        choice_beta_time_pt *= exp(
            beta_time_pt_lambda_environment
            * environmental_attitude.structural_equation_jax
        )

        beta_time_pt_lambda_car_centric = Beta(
            'beta_time_pt_lambda_car_centric', -1, None, 0, 0
        )
        choice_beta_time_pt *= exp(
            beta_time_pt_lambda_car_centric
            * car_centric_attitude.structural_equation_jax
        )

    choice_beta_waiting_time_work = Beta('choice_beta_waiting_time_work', 0, None, 0, 1)
    choice_beta_waiting_time_other_purposes = Beta(
        'choice_beta_waiting_time_other_purposes', 0, None, 0, 1
    )
    choice_beta_waiting_time = (
        choice_beta_waiting_time_work * work_trip
        + choice_beta_waiting_time_other_purposes * other_trip_purposes
    )

    log_scale_choice_model = Beta('log_scale_choice_model', 0, None, None, 1)
    scale_choice_model = exp(log_scale_choice_model)

    # %%
    # Alternative specific constants (kept as-is; they enter only if the LV exists)
    choice_car_centric_car_cte = Beta('choice_car_centric_car_cte', 1, None, None, 0)
    choice_car_centric_pt_cte = Beta('choice_car_centric_pt_cte', 0, None, None, 0)
    choice_environment_car_cte = Beta('choice_environment_car_cte', 0, None, None, 0)
    choice_environment_pt_cte = Beta('choice_environment_pt_cte', 1, None, None, 0)

    # %%
    # Definition of utility functions:
    v_public_transport = scale_choice_model * (
        choice_asc_pt
        + choice_beta_time_pt * TimePT_hour
        + choice_beta_waiting_time * WaitingTimePT / 60
        + choice_beta_cost * MarginalCostPT
        + (
            choice_car_centric_pt_cte * car_centric_attitude.structural_equation_jax
            if car_centric_attitude is not None
            else Numeric(0)
        )
        + (
            choice_environment_pt_cte * environmental_attitude.structural_equation_jax
            if environmental_attitude is not None
            else Numeric(0)
        )
    )

    v_car = scale_choice_model * (
        choice_asc_car
        + choice_beta_time_car * TimeCar_hour
        + choice_beta_cost * CostCarCHF
        + (
            choice_car_centric_car_cte * car_centric_attitude.structural_equation_jax
            if car_centric_attitude is not None
            else Numeric(0)
        )
        + (
            choice_environment_car_cte * environmental_attitude.structural_equation_jax
            if environmental_attitude is not None
            else Numeric(0)
        )
    )

    v_slow_modes = scale_choice_model * (choice_beta_dist * distance_km)

    # %%
    # Associate utility functions with the numbering of alternatives
    v = {0: v_public_transport, 1: v_car, 2: v_slow_modes}

    return v
```

Estimation workflow

These scripts orchestrate model estimation, either by reading existing results or launching new estimation runs, and provide utilities for batch execution.

6.7 estimate.py

```python
from IPython.core.display_functions import display
from biogeme.bayesian_estimation import (
    get_pandas_estimated_parameters as get_pandas_bayesian_estimated_parameters,
)
from biogeme.biogeme import BIOGEME
from biogeme.expressions import Expression, MonteCarlo, log
from biogeme.latent_variables import EstimationMode
from biogeme.models import logit, loglogit
from biogeme.results_processing import (
    get_pandas_estimated_parameters as get_pandas_ml_estimated_parameters,
)

from choice_model import generate_choice_model
from config import (
    Config,
)
from latent_variables import car_name, env_name
from mimic import generate_mimic_model
from optima import Choice, read_data
from read_or_estimate import read_or_estimate


def generate_expression(config: Config) -> Expression:
    utilities = generate_choice_model(config=config)

    # If there are no latent variables, return only the choice model.
    if config.latent_variables == "zero":
        return (
            loglogit(utilities, None, Choice)
            if config.estimation == "bayes"
            else log(MonteCarlo(logit(utilities, None, Choice)))
        )

    mimic = generate_mimic_model(config=config)
    car_centric_attitude = mimic.get_latent_variable(name=car_name)
    environmental_attitude = mimic.get_latent_variable(name=env_name)

    # Build the "inside" of the likelihood once
    if config.estimation == "bayes":
        inner = mimic.log_measurement_equations()
        if config.choice_model == "yes":
            inner = loglogit(utilities, None, Choice) + inner
        return inner

    # ML
    inner = mimic.measurement_equations()
    if config.choice_model == "yes":
        inner = logit(utilities, None, Choice) * inner
    return log(MonteCarlo(inner))


def estimate_model(config: Config) -> None:
    the_expression = generate_expression(config=config)
    estimation_mode = (
        EstimationMode.BAYESIAN
        if config.estimation == "bayes"
        else EstimationMode.MAXIMUM_LIKELIHOOD
    )
    # %%
    # Read the data
    database = read_data()

    # %%
    # Create the Biogeme object
    the_biogeme = BIOGEME(
        database,
        the_expression,
        warmup=config.number_of_bayesian_draws_per_chain,
        bayesian_draws=config.number_of_bayesian_draws_per_chain,
        chains=4,
        number_of_draws=config.number_of_monte_carlo_draws,
        calculating_second_derivatives='never',
        numerically_safe=True,
        max_iterations=5000,
    )
    the_biogeme.model_name = config.name

    # %%
    # If estimation results are saved on file, we read them to speed up the process.
    # If not, we estimate the parameters.
    results = read_or_estimate(
        the_biogeme=the_biogeme,
        estimation_mode=estimation_mode,
        directory='saved_results',
    )

    # %%
    print(results.short_summary())

    # %%
    # Get the results in a pandas table
    pandas_results = (
        get_pandas_ml_estimated_parameters(
            estimation_results=results,
        )
        if estimation_mode == EstimationMode.MAXIMUM_LIKELIHOOD
        else get_pandas_bayesian_estimated_parameters(estimation_results=results)
    )
    display(pandas_results)
```

6.8 read_or_estimate.py

```
"""

Read of estimate
================

Function to estimate the parameters, or read them from a file, if available.

Michel Bierlaire, EPFL
Mon May 05 2025, 18:59:34
"""

from biogeme.bayesian_estimation import BayesianResults
from biogeme.biogeme import BIOGEME
from biogeme.latent_variables import EstimationMode
from biogeme.results_processing import EstimationResults


def read_or_estimate(
    the_biogeme: BIOGEME, estimation_mode: EstimationMode, directory: str = '.'
) -> EstimationResults | BayesianResults:
    """
    Function to estimate the parameters, or read them from a file, if available.

    :param the_biogeme: Biogeme object.
    :param estimation_mode: EstimationMode.BAYESIAN or EstimationMode.MAXIMUM_LIKELIHOOD
    :param directory: directory where the yaml file is supposed to be.

    :return: estimation results.
    """
    if estimation_mode == EstimationMode.BAYESIAN:
        try:
            filename = f'{directory}/{the_biogeme.model_name}.nc'
            results = BayesianResults.from_netcdf(filename=filename)
            print(f'Results are read from the file {filename}.')
        except FileNotFoundError:
            print('Parameters are being estimated.')
            results = the_biogeme.bayesian_estimation()
        return results

    if estimation_mode != EstimationMode.MAXIMUM_LIKELIHOOD:
        raise ValueError(f'Unknown estimation mode: {estimation_mode}')

    try:
        filename = f'{directory}/{the_biogeme.model_name}.yaml'
        results = EstimationResults.from_yaml_file(filename=filename)
        print(f'Results are read from the file {filename}.')
    except FileNotFoundError:
        print('Parameters are being estimated.')
        results = the_biogeme.estimate()
    return results
```

6.9 plot_b01_choice_only_ml.py

```
"""Choice model only. Maximum likelihood estimation

Michel Bierlaire
Tue Dec 23 2025, 14:52:48

"""

import biogeme.biogeme_logging as blog

from config import Config
from estimate import estimate_model

logger = blog.get_screen_logger(level=blog.INFO)

# Choice model only

the_config = Config(
    name='b01_choice_only_ml',
    latent_variables="zero",
    choice_model="yes",
    estimation="ml",
    number_of_bayesian_draws_per_chain=20_000,
    number_of_monte_carlo_draws=20_000,
)

estimate_model(config=the_config)
```

6.10 plot_b02_mimic_ml.py

```
"""MIMIC model. Maximum likelihood estimation

Michel Bierlaire
Tue Dec 23 2025, 14:53:49

"""

import biogeme.biogeme_logging as blog

from config import Config
from estimate import estimate_model

logger = blog.get_screen_logger(level=blog.INFO)

the_config = Config(
    name='b02_mimic_ml',
    latent_variables="two",
    choice_model="no",
    estimation="ml",
    number_of_bayesian_draws_per_chain=20_000,
    number_of_monte_carlo_draws=20_000,
)

estimate_model(config=the_config)
```

6.11 plot_b03_hybrid_ml.py

```
"""Hybrid choice model. Maximum likelihood estimation

Michel Bierlaire
Tue Dec 23 2025, 14:57:15
"""

import biogeme.biogeme_logging as blog

from config import Config
from estimate import estimate_model

logger = blog.get_screen_logger(level=blog.INFO)

the_config = Config(
    name='b03_hybrid_ml',
    latent_variables="two",
    choice_model="yes",
```

```python
    estimation="ml",
    number_of_bayesian_draws_per_chain=20_000,
    number_of_monte_carlo_draws=20_000,
)

estimate_model(config=the_config)
```

6.12 plot_b04_choice_only_bayes.py

```python
"""Choice model only. Bayesian estimation

Michel Bierlaire
Tue Dec 23 2025, 14:56:09

"""

import biogeme.biogeme_logging as blog

from config import Config
from estimate import estimate_model

logger = blog.get_screen_logger(level=blog.INFO)

the_config = Config(
    name='b04_choice_only_bayes',
    latent_variables="zero",
    choice_model="yes",
    estimation="bayes",
    number_of_bayesian_draws_per_chain=20_000,
    number_of_monte_carlo_draws=20_000,
)

estimate_model(config=the_config)
```

6.13 plot_b05_mimic_bayes.py

```python
"""MIMIC model. Bayesian estimation

Michel Bierlaire
Tue Dec 23 2025, 14:56:34
"""

import biogeme.biogeme_logging as blog

from config import Config
from estimate import estimate_model

logger = blog.get_screen_logger(level=blog.INFO)

the_config = Config(
    name='b05_mimic_bayes',
    latent_variables="two",
    choice_model="no",
    estimation="bayes",
    number_of_bayesian_draws_per_chain=20_000,
    number_of_monte_carlo_draws=20_000,
)

estimate_model(config=the_config)
```

6.14 plot_b06_hybrid_bayes.py

```python
"""Hybrid choice model. Bayesian estimation

Michel Bierlaire
Tue Dec 23 2025, 14:57:15
"""

import biogeme.biogeme_logging as blog

from config import Config
from estimate import estimate_model

logger = blog.get_screen_logger(level=blog.INFO)

the_config = Config(
    name='b06_hybrid_bayes',
    latent_variables="two",
    choice_model="yes",
    estimation="bayes",
    number_of_bayesian_draws_per_chain=20_000,
    number_of_monte_carlo_draws=20_000,
)

estimate_model(config=the_config)
```

References

Ashok, K., Dillon, W. R. and Yuan, S. (2002). Extending discrete choice models to incorporate attitudinal and other latent variables, *Journal of Marketing Research* **39**(1): 31–46.

Ben-Akiva, M., Walker, J., Bernardino, A. T., Gopinath, D. A., Morikawa, T. and Polydoropoulou, A. (2002). Integration of choice and latent variable models, *Perpetual motion: Travel behaviour research opportunities and application challenges* pp. 431–470.

Bierlaire, M. (2019). Monte-carlo integration with pandasbiogeme, *Technical Report TRANSP-OR 191231*, Lausanne, Switzerland.

Bierlaire, M., Curchod, A., Danalet, A., Doyen, E., Faure, P., Glerum, A., Kaufmann, V., Tabaka, K. and Schuler, M. (2011). Projet de recherche sur la mobilité combinée, rapport définitif de l'enquête de préférences révélées, *Technical Report TRANSP-OR 110704*, Transport and Mobility Laboratory, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland.

Greene, W. H. and Hensher, D. A. (2003). A latent class model for discrete choice analysis: contrasts with mixed logit, *Transportation Research Part B: Methodological* **37**(8): 681–698.

Likert, R. (1932). A technique for the measurement of attitudes, *Archives of psychology* **140**: 1–55.

Walker, J. L. (2001). *Extended discrete choice models: integrated framework, flexible error structures, and latent variables*, PhD thesis, Massachusetts Institute of Technology.