Estimating hybrid choice models with Biogeme

Michel Bierlaire   Moshe Ben-Akiva   Joan Walker

December 25, 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:

$$\begin{aligned} \tau_1 &= -\delta_0 - \delta_1, \\ \tau_2 &= -\delta_0, \\ \tau_3 &= \quad \delta_0, \\ \tau_4 &= \quad \delta_0 + \delta_1, \end{aligned}$$

where $\delta_0 > 0$ and $\delta_1 > 0$ 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.

2 Normalization and identification in latent-variable models

Models with latent variables contain parameters that are not uniquely identified from the data unless additional restrictions are imposed. These restrictions, referred to as *normalizations*, are not behavioral assumptions; rather, they fix the arbitrary units of measurement (location and scale) that are inherent to latent constructs and ordinal measurement schemes.

This section explains why normalization is required, where non-identification arises, and how to impose a consistent and non-redundant set of normalizations. We distinguish carefully between normalizations related to the *latent variables themselves* and those related to the *ordinal measurement models.* Although the reference-indicator strategy is emphasized for its interpretability and numerical stability, the underlying principles apply more generally.

2.1 Sources of non-identification

Latent-variable models are invariant to specific transformations of the latent variables and associated parameters. Without normalization, multiple parameter vectors generate exactly the same likelihood. In such a case, estimation is ill-posed: the likelihood exhibits flat or weakly curved directions, standard errors are unreliable, and numerical optimization or Bayesian sampling may be unstable.

Two fundamental sources of non-identification arise at the level of the latent variables:

- **Location invariance**: the origin of a latent variable is arbitrary.
- **Scale invariance**: the unit in which a latent variable is measured is arbitrary.

Both must be addressed for each latent variable in the model. Note that the need for normalization in latent-variable models is closely analogous to the well-known identification issues in logit models and related random utility models. In logit, utilities are latent and only differences in utility matter. As a consequence, utilities are invariant to the addition of a constant (location invariance) and to multiplication by a positive scalar (scale invariance). Standard practice therefore requires fixing one alternative-specific constant (or equivalently normalizing utilities relative to a base alternative) and fixing the scale of utility, typically by normalizing the variance of the error term.

Latent variables play a role analogous to latent utilities: they enter the likelihood only through systematic components and relative comparisons.

Continuous measurement equations therefore require exactly the same type of normalization as utility functions in logit models — one restriction to fix location and one to fix scale. Likewise, ordinal measurement equations introduce additional latent constructs (latent responses) that must be normalized in the same spirit as discrete choice utilities defined over ordered alternatives.

2.2 Normalization for continuous measurement equations

Consider the linear measurement and structural equations introduced in Section 1. If intercepts are estimated, any shift of a latent variable can be absorbed by a compensating shift in the measurement intercepts. Likewise, any rescaling of a latent variable can be offset by rescaling loadings and structural parameters. As a result, neither the location nor the scale of the latent variable is identified from the data alone.

Therefore, for each latent variable, exactly two normalizations are required:

- one to fix its *location*;
- one to fix its *scale*.

A practical and interpretable approach is the *reference-indicator strategy*. For each latent variable $\mathtt{k}$, one indicator $\ell(\mathtt{k})$ is selected as its reference indicator. Location is fixed by setting the corresponding intercept to zero, and scale is fixed by fixing the corresponding loading to ± 1. The sign choice is a convention that determines the orientation of the latent scale and should be chosen to preserve semantic consistency between the latent variable and its indicators.

Once these two constraints are imposed, remaining parameters—such as measurement error variances—become meaningful and interpretable.

2.3 Ordinal indicators and ordered probit measurement models

We now consider ordinal indicators modeled using an ordered probit specification, as introduced in Section 1.3. In this case, the observed response does not reveal the latent measurement equation directly, but only the interval in which an underlying latent response lies.

Compared to continuous indicators, ordered probit models introduce *additional* invariances:

- a **threshold-location invariance**: adding the same constant to the latent response and all thresholds leaves choice probabilities unchanged;
- a **threshold-scale invariance**: multiplying the latent response, all thresholds, and the measurement error standard deviation by the same positive constant leaves probabilities unchanged.

These invariances are *specific to each ordered probit measurement equation.* Crucially, they arise *independently for each distinct set of thresholds.* As a consequence, normalization of the ordered probit layer must be performed *separately for each independent threshold system.*

2.4 Separation of roles: latent variables vs. ordinal layers

It is essential to distinguish clearly between:

- normalizations that identify the *latent variables*;
- normalizations that identify the *ordinal measurement scales.*

The reference-indicator normalizations (intercept and loading) serve only to fix the location and scale of the latent variables. They do *not* resolve the invariances intrinsic to the ordered probit measurement model. Those must be addressed at the level of the ordinal layer itself.

2.5 Normalization procedure for ordinal indicators

For each ordinal indicator (or group of indicators) that shares a common set of thresholds, the following steps are required.

Step 1: Anchor the latent variable (once per latent variable). If the ordinal indicator is chosen as the reference indicator for a latent variable, impose the same normalization as in the continuous case: fix the intercept and one loading. This step fixes the location, scale, and orientation of the latent variable and should be applied *once per latent variable*, not per indicator.

Step 2: Fix the location of the threshold system (once per threshold set). Because only differences between the latent response and the thresholds matter, the threshold system has an arbitrary origin. This must be fixed exactly once for each independent set of thresholds. This can be done either by fixing one threshold to zero or, when the response scale is designed to be symmetric, by imposing symmetry around zero. The latter approach is often preferable, as it aligns the statistical parameterization with the semantics of Likert-type scales and reduces dimensionality.

Step 3: Fix the scale of the ordered probit layer (once per threshold set). Ordered probit models do not identify the scale of the latent response relative to the threshold spacing and the measurement error variance. Therefore, one scale normalization is required for each independent threshold system. A standard and convenient choice is to fix the measurement error standard deviation to one. This follows conventional ordered probit practice and prevents threshold spacings from being confounded with noise variance.

2.6 Key principle: no redundancy

Each source of invariance must be removed *once, and only once.* Overnormalization—such as fixing both threshold location and an equivalent intercept, or fixing both loading and latent variance—does not improve identification and may instead induce artificial parameter correlations or numerical instability.

2.7 Alternative normalizations and choice of parameterization

The normalizations described above are not unique. As in discrete choice models, several equivalent normalization strategies can be adopted, leading to identical likelihood values and identical fitted probabilities. The choice among them affects interpretation, numerical stability, and the transparency of the resulting parameters, but not the model's ability to fit the data.

For latent variables, scale can be fixed in different ways. Instead of fixing a factor loading to ± 1, one may fix the variance of the latent variable (or of the structural disturbance) to one, or impose a "money-metric" normalization by fixing the coefficient of a monetary variable to a known value. Such approaches are common in structural discrete choice models and welfare analysis. However, when latent variables are measured through multiple indicators, these alternatives often lead to less transparent interpretations: the latent variable is no longer anchored directly to an observed indicator, and factor loadings must be interpreted relative to an abstract latent scale.

Similarly, in ordered probit measurement models, scale may in principle be fixed by normalizing threshold spacings or by normalizing the variance of the latent response. Fixing the measurement error variance to one is the most common convention, as it cleanly separates threshold locations from noise and aligns the specification with standard ordered probit practice.

In this work, we adopt the reference-indicator strategy for latent variables and variance normalization for ordered probit layers because this combination offers three practical advantages. First, it yields parameters with a direct and intuitive interpretation: the latent variable is measured in the same units as a concrete, observed indicator. Second, it avoids entangling scale normalization with structural parameters, which improves numerical stability in both maximum likelihood and Bayesian estimation. Third, it allows different ordinal indicators — possibly with different numbers of categories or symmetric designs — to be normalized independently and consistently.

Importantly, this choice reflects a modeling convention rather than a substantive behavioral assumption. Alternative normalizations are valid and may be preferable in other contexts, provided that each source of invariance is addressed exactly once and that the resulting parameter interpretation is clearly documented.

2.8 Summary of recommended practice

The main normalization principles are summarized in Table 1.

- For each latent variable: fix one intercept and one loading using a reference indicator.
- For each independent ordered probit threshold system:
  - fix threshold location (preferably via symmetry);
  - fix threshold scale (e.g. by setting the measurement error standard deviation to one).

This strategy removes all location and scale indeterminacies without redundancy, yields stable estimation, and ensures that all parameters retain a clear and interpretable meaning.

3 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*.

Table 1: Summary of normalization requirements in latent-variable and ordered probit models

Model component	**Source of non-identification**	**Normalization required**	**Applied how often**
Latent variable x_k^*	Location invariance	Fix one measurement intercept (reference indicator)	Once per latent variable
Latent variable x_k^*	Scale invariance	Fix one loading to ± 1 (reference indicator)	Once per latent variable
Continuous measurement equation	None beyond latent-variable invariance	No additional normalization	—
Ordinal measurement (ordered probit)	Threshold location invariance	Fix one threshold to zero *or* impose symmetry	Once per independent threshold set
Ordinal measurement (ordered probit)	Threshold scale invariance	Fix measurement error scale (e.g. $\sigma = 1$)	Once per independent threshold set
Multiple ordinal indicators	Independent threshold systems	Each threshold system normalized separately	Once per threshold system

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.

3.1 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:

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

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:

$$L^{\text{MIMIC}}_n = \int_{x^*_n} L^I_n(I_n \mid x^*_n, x_n)\; f(x^*_n \mid x_n)\; dx^*_n, \tag{16}$$

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

3.2 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):

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

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

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

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

4 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 7.

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.

4.1 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.

4.2 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.

4.3 Measurement equations

For each individual n and each indicator ℓ described in Section 4.1, 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 2 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:

$$\begin{aligned}
\tau_1 &= -(\delta_0 + \delta_1), \\
\tau_2 &= -\delta_0, \\
\tau_3 &= \delta_0, \\
\tau_4 &= (\delta_0 + \delta_1),
\end{aligned}$$

where $\delta_0 > 0$ and $\delta_1 > 0$ 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.

4.4 Implementation notes

The results reported below are produced using the set of Python specification files included in the appendix (Sections 8.1–8.14). 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 (8.1). The file in Section 8.1 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 (8.4). The file in Section 8.4 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 (8.3). The file in Section 8.3 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 (8.2). To ensure that all model variants are generated from the same building blocks, the implementation relies on a configuration object defined in Section 8.2. 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 (8.5). The file in Section 8.5 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 (8.6). The file in Section 8.6 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 (8.8). The file in Section 8.8 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 (8.7). Finally, the file in Section 8.7 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 (8.9–8.14). 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 8.2) and then relies on the generic estimation workflow (Sections 8.8 and 8.7) to either run the estimation or read existing outputs.

- **Choice-only, maximum likelihood** (Section 8.9): 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 8.10): 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 8.11): 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 8.12): 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 8.13): 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 8.14): 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 8.3, 8.4, 8.5, and 8.6); the differences between them arise solely from the configuration settings and the chosen estimation paradigm.

5 Estimation results

In this section, we report the estimation results for each of the three models:

- choice model: general statistics in Table 2 and estimated parameters in Table 3 on page 26;
- MIMIC model: general statistics in Table 4 on page 26, estimated parameters of the structural equations in Table 5 on page 27, and estimated parameters of the measurement equations in Table 6 on page 24;
- Hybrid choice model: general statistice in Table 7 on page 27, estimated parameters of the choice model in Table 8 on page 28, estimated parameters of the structural equations in Table 9 on page 29, and estimated parameters of the measurement equations in Table 10 on page 25.

Number of estimated parameters	7
Sample size	896
Init log likelihood	-984.3566
Final log likelihood	-512.5193
Likelihood ratio test for the init. model	943.6747
Rho-square for the init. model	0.479
Rho-square-bar for the init. model	0.472
Akaike Information Criterion	1039.039
Bayesian Information Criterion	1072.624

Table 2: Choice model: general statistics

Parameter number	Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
29	measurement_coefficient_environmental_attitude_Envir01	1.74	0.457	3.8	0.000142

30	measurement_coefficient_car_centric_attitude_Envir02	-0.201	0.0383	-5.24	1.6e-07
31	measurement_Envir02_sigma_log	-1.07	0.221	-4.86	1.19e-06
32	measurement_intercept_Envir03	0.664	0.184	3.62	0.000294
33	measurement_coefficient_environmental_attitude_Envir03	-1.17	0.199	-5.89	3.84e-09
34	measurement_Envir03_sigma_log	-0.38	0.242	-1.57	0.116
35	measurement_intercept_Envir04	0.185	0.0597	3.09	0.00198
36	measurement_coefficient_environmental_attitude_Envir04	-0.773	0.0943	-8.19	2.22e-16
37	measurement_Envir04_sigma_log	-0.995	0.222	-4.48	7.41e-06
38	measurement_intercept_Envir05	0.128	0.0533	2.41	0.016
39	measurement_coefficient_environmental_attitude_Envir05	1.11	0.113	9.82	0.0
40	measurement_Envir05_sigma_log	-1.05	0.229	-4.61	4.06e-06
41	measurement_intercept_Envir06	0.277	0.0655	4.23	2.33e-05
42	measurement_coefficient_car_centric_attitude_Envir06	-0.0436	0.0202	-2.16	0.0309
43	measurement_coefficient_environmental_attitude_Envir06	0.91	0.0847	10.7	0.0
44	measurement_Envir06_sigma_log	-1.63	0.227	-7.16	8.21e-13
45	measurement_intercept_Mobil03	0.479	0.139	3.44	0.000591
46	measurement_coefficient_car_centric_attitude_Mobil03	-0.209	0.0539	-3.88	0.000103
47	measurement_Mobil03_sigma_log	-0.445	0.244	-1.83	0.0675
48	measurement_intercept_Mobil05	0.786	0.266	2.95	0.00314
49	measurement_coefficient_car_centric_attitude_Mobil05	-0.416	0.104	-3.99	6.61e-05
50	measurement_Mobil05_sigma_log	-0.107	0.28	-0.381	0.703
51	measurement_intercept_Mobil08	0.0853	0.069	1.24	0.217
52	measurement_coefficient_car_centric_attitude_Mobil08	0.28	0.0782	3.58	0.000345
53	measurement_Mobil08_sigma_log	-0.127	0.268	-0.473	0.636
54	measurement_intercept_Mobil09	0.67	0.171	3.91	9.28e-05
55	measurement_coefficient_car_centric_attitude_Mobil09	-0.256	0.0509	-5.03	4.85e-07
56	measurement_Mobil09_sigma_log	-0.842	0.232	-3.63	0.000286
57	measurement_intercept_Mobil10	1.22	0.637	1.91	0.056
58	measurement_coefficient_car_centric_attitude_Mobil10	1.17	0.464	2.53	0.0115
59	measurement_Mobil10_sigma_log	0.464	0.473	0.981	0.327
60	measurement_intercept_Mobil12	0.636	0.462	1.38	0.169
61	measurement_coefficient_environmental_attitude_Mobil12	-3.34	1.51	-2.21	0.0273
62	measurement_Mobil12_sigma_log	0.775	0.481	1.61	0.107
63	measurement_intercept_LifSty01	0.0742	0.0475	1.56	0.118
64	measurement_coefficient_environmental_attitude_LifSty01	0.342	0.092	3.72	0.000197

65	measurement_LifSty01_sigma_log	-0.62	0.231	-2.69	0.00719
66	measurement_intercept_LifSty07	-0.0521	0.0535	-0.974	0.33
67	measurement_coefficient_car_centric_attitude_LifSty07	0.0896	0.0449	2.0	0.0458
68	measurement_LifSty07_sigma_log	-0.208	0.274	-0.76	0.447
69	$\ln(\delta_0)$	-2.61	0.223	-11.7	0.0
70	$\ln(\delta_1)$	-1.03	0.21	-4.9	9.66e-07
71	measurement_intercept_NbCar	2.06	0.139	14.8	0.0
72	measurement_coefficient_car_centric_attitude_NbCar	0.67	0.111	6.01	1.84e-09
73	$\ln(\delta_1^{\text{NbCar}})$	0.67	0.0714	9.39	0.0
74	$\ln(\delta_2^{\text{NbCar}})$	0.614	0.0664	9.26	0.0

Table 6: MIMIC model: estimated parameters of the measurement equations

Parameter number	Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
29	measurement_coefficient_environmental_attitude_Envir01	1.84	0.428	4.31	1.66e-05
30	measurement_coefficient_car_centric_attitude_Envir02	-0.183	0.0351	-5.21	1.9e-07
31	measurement_Envir02_sigma_log	-1.03	0.206	-4.99	5.98e-07
32	measurement_intercept_Envir03	0.689	0.181	3.8	0.000144
33	measurement_coefficient_environmental_attitude_Envir03	-1.09	0.181	-6.02	1.78e-09
34	measurement_Envir03_sigma_log	-0.366	0.231	-1.58	0.113
35	measurement_intercept_Envir04	0.198	0.0586	3.38	0.000737
36	measurement_coefficient_environmental_attitude_Envir04	-0.706	0.0853	-8.27	2.22e-16
37	measurement_Envir04_sigma_log	-0.958	0.211	-4.54	5.53e-06
38	measurement_intercept_Envir05	0.144	0.0579	2.48	0.0131
39	measurement_coefficient_environmental_attitude_Envir05	1.02	0.105	9.71	0.0
40	measurement_Envir05_sigma_log	-0.985	0.217	-4.54	5.53e-06
41	measurement_intercept_Envir06	0.315	0.0737	4.27	1.95e-05
42	measurement_coefficient_car_centric_attitude_Envir06	-0.0627	0.0232	-2.71	0.00681
43	measurement_coefficient_environmental_attitude_Envir06	0.814	0.0764	10.7	0.0
44	measurement_Envir06_sigma_log	-1.53	0.213	-7.16	8.01e-13
45	measurement_intercept_Mobil03	0.49	0.137	3.57	0.000357
46	measurement_coefficient_car_centric_attitude_Mobil03	-0.253	0.0611	-4.14	3.43e-05

47	measurement_Mobil03_sigma_log	-0.471	0.232	-2.03	0.0424
48	measurement_intercept_Mobil05	0.829	0.264	3.14	0.00172
49	measurement_coefficient_car_centric_attitude_Mobil05	-0.457	0.111	-4.11	4.03e-05
50	measurement_Mobil05_sigma_log	-0.119	0.266	-0.446	0.656
51	measurement_intercept_Mobil08	0.0849	0.0768	1.11	0.269
52	measurement_coefficient_car_centric_attitude_Mobil08	0.238	0.0799	2.98	0.00286
53	measurement_Mobil08_sigma_log	-0.027	0.267	-0.101	0.92
54	measurement_intercept_Mobil09	0.775	0.184	4.22	2.47e-05
55	measurement_coefficient_car_centric_attitude_Mobil09	-0.251	0.0515	-4.87	1.09e-06
56	measurement_Mobil09_sigma_log	-0.741	0.219	-3.38	0.000738
57	measurement_intercept_Mobil10	1.35	0.814	1.66	0.0964
58	measurement_coefficient_car_centric_attitude_Mobil10	1.39	0.633	2.19	0.0286
59	measurement_Mobil10_sigma_log	0.669	0.529	1.26	0.206
60	measurement_intercept_Mobil12	0.676	0.463	1.46	0.144
61	measurement_coefficient_environmental_attitude_Mobil12	-3.06	1.34	-2.29	0.022
62	measurement_Mobil12_sigma_log	0.76	0.463	1.64	0.101
63	measurement_intercept_LifSty01	0.0613	0.0476	1.29	0.199
64	measurement_coefficient_environmental_attitude_LifSty01	0.339	0.0857	3.96	7.58e-05
65	measurement_LifSty01_sigma_log	-0.603	0.218	-2.77	0.00561
66	measurement_intercept_LifSty07	-0.0608	0.0547	-1.11	0.266
67	measurement_coefficient_car_centric_attitude_LifSty07	0.0723	0.0483	1.5	0.135
68	measurement_LifSty07_sigma_log	-0.176	0.265	-0.663	0.507
69	likert_delta_0_log	-2.6	0.21	-12.3	0.0
70	likert_delta_1_log	-1.0	0.197	-5.09	3.54e-07
71	measurement_intercept_NbCar	2.04	0.149	13.8	0.0
72	measurement_coefficient_car_centric_attitude_NbCar	0.816	0.122	6.68	2.45e-11
73	cars_delta_1_log	0.719	0.0711	10.1	0.0
74	cars_delta_2_log	0.663	0.0644	10.3	0.0

Table 10: Hybrid choice model: estimated parameters of the measurement equations

Parameter number	Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
0	choice_asc_pt	-0.665	0.277	-2.4	0.0163
1	choice_beta_time_pt_ref	-0.956	0.113	-8.44	0.0
2	choice_beta_cost	-0.0795	0.00833	-9.55	0.0
3	choice_asc_car	-0.246	0.231	-1.07	0.286
4	choice_beta_time_car_ref	-2.03	0.155	-13.1	0.0
5	choice_beta_dist_work	-0.205	0.0148	-13.8	0.0
6	choice_beta_dist_other_purposes	-0.324	0.0306	-10.6	0.0

Table 3: Choice model: estimated parameters

Number of estimated parameters	60
Sample size	896
Init log likelihood	-29507.39
Final log likelihood	-17009.57
Likelihood ratio test for the init. model	24995.64
Rho-square for the init. model	0.424
Rho-square-bar for the init. model	0.422
Akaike Information Criterion	34139.14
Bayesian Information Criterion	34427.01
Number of draws	20000

Table 4: MIMIC model: general statistics

Parameter number	Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
15	struct_environmental_attitude_childSuburb	0.0773	0.0306	2.53	0.0115
16	struct_environmental_attitude_ScaledIncome	0.0195	0.00577	3.39	0.000711
17	struct_environmental_attitude_city_center_as_kid	0.151	0.0731	2.07	0.0389
18	struct_environmental_attitude_artisans	0.129	0.0588	2.19	0.0288
19	struct_environmental_attitude_high_education	0.14	0.0461	3.04	0.00237
20	struct_environmental_attitude_low_education	0.045	0.0306	1.47	0.142
21	struct_environmental_attitude_sigma_log	-1.32	0.232	-5.7	1.22e-08
22	struct_car_centric_attitude_high_education	-0.451	0.106	-4.25	2.1e-05
23	struct_car_centric_attitude_top_manager	0.146	0.135	1.08	0.28
24	struct_car_centric_attitude_employees	-0.0609	0.0813	-0.748	0.454
25	struct_car_centric_attitude_age_30_less	0.489	0.172	2.85	0.00442
26	struct_car_centric_attitude_ScaledIncome	0.00453	0.0109	0.416	0.677
27	struct_car_centric_attitude_car_oriented_parents	0.324	0.0998	3.25	0.00116
28	struct_car_centric_attitude_sigma_log	-0.17	0.102	-1.67	0.0949

Table 5: MIMIC model: estimated parameters of the structural equations

Number of estimated parameters	75
Sample size	896
Excluded observations	0
Init log likelihood	-29927.74
Final log likelihood	-17432.33
Likelihood ratio test for the init. model	24990.82
Rho-square for the init. model	0.418
Rho-square-bar for the init. model	0.415
Akaike Information Criterion	35014.65
Bayesian Information Criterion	35374.5
Final gradient norm	4.7392E-01
Number of draws	20000

Table 7: Hybrid choice model: general statistics

Parameter number	Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
0	choice_asc_pt	-2.1	0.725	-2.9	0.00373
1	choice_beta_time_pt_ref	-1.59	0.397	-4.0	6.24e-05
2	choice_beta_cost	-0.104	0.0132	-7.86	3.77e-15
3	choice_asc_car	-1.62	0.615	-2.63	0.00856
4	choice_beta_time_car_ref	-3.55	0.943	-3.76	0.000167
5	choice_beta_dist_work	-0.597	0.0809	-7.38	1.54e-13
6	choice_beta_dist_other_purposes	-0.898	0.163	-5.5	3.87e-08
7	beta_time_pt_lambda_ environment	-0.817	0.407	-2.01	0.0446
8	beta_time_pt_lambda_car_centric	-1.32	0.213	-6.21	5.15e-10
9	choice_car_centric_pt_cte	-0.676	0.673	-1.0	0.316
10	choice_environment_pt_cte	-0.42	1.47	-0.286	0.775
11	beta_time_car_lambda_ environment	-1.01	0.451	-2.23	0.0259
12	beta_time_car_lambda_car_ centric	-1.58	0.251	-6.31	2.7e-10
13	choice_car_centric_car_cte	1.06	0.622	1.71	0.0872
14	choice_environment_car_cte	-0.483	1.24	-0.39	0.697

Table 8: Hybrid choice model: estimated parameters of the choice model

Parameter number	Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
15	struct_environmental_attitude_childSuburb	0.0857	0.0329	2.6	0.00923
16	struct_environmental_attitude_ScaledIncome	0.0228	0.00603	3.77	0.00016
17	struct_environmental_attitude_city_center_as_kid	0.169	0.084	2.01	0.044
18	struct_environmental_attitude_artisans	0.136	0.0622	2.19	0.0282
19	struct_environmental_attitude_high_education	0.161	0.0499	3.22	0.00127
20	struct_environmental_attitude_low_education	0.0512	0.0339	1.51	0.13
21	struct_environmental_attitude_sigma_log	-1.23	0.213	-5.77	7.74e-09
22	struct_car_centric_attitude_high_education	-0.396	0.099	-4.0	6.33e-05
23	struct_car_centric_attitude_top_manager	0.199	0.134	1.48	0.139
24	struct_car_centric_attitude_employees	-0.0551	0.0771	-0.714	0.475
25	struct_car_centric_attitude_age_30_less	0.437	0.162	2.69	0.00716
26	struct_car_centric_attitude_ScaledIncome	0.0171	0.0108	1.58	0.113
27	struct_car_centric_attitude_car_oriented_parents	0.312	0.0919	3.39	0.000692
28	struct_car_centric_attitude_sigma_log	-0.228	0.102	-2.24	0.025

Table 9: Hybrid choice model: estimated parameters of the structural equations

5.1 Comparison of the three models

We report below the value of each parameter in each model.

	Parameter name	Choice only Coef./(SE)	MIMIC Coef./(SE)	Hybrid Coef./(SE)
0	beta_time_car_lambda_car_centric			-1.58^{***} (0.251)
1	beta_time_car_lambda_environment			-1.01^{**} (0.451)
2	beta_time_pt_lambda_car_centric			-1.32^{***} (0.213)
3	beta_time_pt_lambda_environment			-0.817^{**} (0.407)
4	cars_delta_1_log		0.67^{***} (0.0714)	0.719^{***} (0.0711)
5	cars_delta_2_log		0.614^{***} (0.0664)	0.663^{***} (0.0644)
6	choice_asc_car	-0.246 (0.231)		-1.62^{***} (0.615)
7	choice_asc_pt	-0.665^{**} (0.277)		-2.1^{***} (0.725)
8	choice_beta_cost	-0.0795^{***} (0.00833)		-0.104^{***} (0.0132)
9	choice_beta_dist_other_purposes	-0.324^{***} (0.0306)		-0.898^{***} (0.163)
10	choice_beta_dist_work	-0.205^{***} (0.0148)		-0.597^{***} (0.0809)
11	choice_beta_time_car_ref	-2.03^{***} (0.155)		-3.55^{***} (0.943)
12	choice_beta_time_pt_ref	-0.956^{***} (0.113)		-1.59^{***} (0.397)
13	choice_car_centric_car_cte			1.06^{*} (0.622)
14	choice_car_centric_pt_cte			-0.676 (0.673)
15	choice_environment_car_cte			-0.483 (1.24)
16	choice_environment_pt_cte			-0.42 (1.47)
17	likert_delta_0_log		-2.61^{***} (0.223)	-2.6^{***} (0.21)
18	likert_delta_1_log		-1.03^{***} (0.21)	-1.0^{***} (0.197)
19	measurement_Envir02_sigma_log		-1.07^{***} (0.221)	-1.03^{***} (0.206)

20	measurement_Envir03_sigma_log	−0.38	−0.366
		(0.242)	(0.231)
21	measurement_Envir04_sigma_log	-0.995^{***}	-0.958^{***}
		(0.222)	(0.211)
22	measurement_Envir05_sigma_log	-1.05^{***}	-0.985^{***}
		(0.229)	(0.217)
23	measurement_Envir06_sigma_log	-1.63^{***}	-1.53^{***}
		(0.227)	(0.213)
24	measurement_LifSty01_sigma_log	-0.62^{***}	-0.603^{***}
		(0.231)	(0.218)
25	measurement_LifSty07_sigma_log	−0.208	−0.176
		(0.274)	(0.265)
26	measurement_Mobil03_sigma_log	-0.445^{*}	-0.471^{**}
		(0.244)	(0.232)
27	measurement_Mobil05_sigma_log	−0.107	−0.119
		(0.28)	(0.266)
28	measurement_Mobil08_sigma_log	−0.127	−0.027
		(0.268)	(0.267)
29	measurement_Mobil09_sigma_log	-0.842^{***}	-0.741^{***}
		(0.232)	(0.219)
30	measurement_Mobil10_sigma_log	0.464	0.669
		(0.473)	(0.529)
31	measurement_Mobil12_sigma_log	0.775	0.76
		(0.481)	(0.463)
32	measurement_coefficient_car_centric_attitude_Envir02	-0.201^{***}	-0.183^{***}
		(0.0383)	(0.0351)
33	measurement_coefficient_car_centric_attitude_Envir06	-0.0436^{**}	-0.0627^{***}
		(0.0202)	(0.0232)
34	measurement_coefficient_car_centric_attitude_LifSty07	0.0896^{**}	0.0723
		(0.0449)	(0.0483)
35	measurement_coefficient_car_centric_attitude_Mobil03	-0.209^{***}	-0.253^{***}
		(0.0539)	(0.0611)
36	measurement_coefficient_car_centric_attitude_Mobil05	-0.416^{***}	-0.457^{***}
		(0.104)	(0.111)
37	measurement_coefficient_car_centric_attitude_Mobil08	0.28^{***}	0.238^{***}
		(0.0782)	(0.0799)
38	measurement_coefficient_car_centric_attitude_Mobil09	-0.256^{***}	-0.251^{***}
		(0.0509)	(0.0515)

39	measurement_coefficient_car_centric_attitude_Mobil10	1.17^{**}	1.39^{**}
		(0.464)	(0.633)
40	measurement_coefficient_car_centric_attitude_NbCar	0.67^{***}	0.816^{***}
		(0.111)	(0.122)
41	measurement_coefficient_environmental_attitude_Envir01	1.74^{***}	1.84^{***}
		(0.457)	(0.428)
42	measurement_coefficient_environmental_attitude_Envir03	-1.17^{***}	-1.09^{***}
		(0.199)	(0.181)
43	measurement_coefficient_environmental_attitude_Envir04	-0.773^{***}	-0.706^{***}
		(0.0943)	(0.0853)
44	measurement_coefficient_environmental_attitude_Envir05	1.11^{***}	1.02^{***}
		(0.113)	(0.105)
45	measurement_coefficient_environmental_attitude_Envir06	0.91^{***}	0.814^{***}
		(0.0847)	(0.0764)
46	measurement_coefficient_environmental_attitude_LifSty01	0.342^{***}	0.339^{***}
		(0.092)	(0.0857)
47	measurement_coefficient_environmental_attitude_Mobil12	-3.34^{**}	-3.06^{**}
		(1.51)	(1.34)
48	measurement_intercept_Envir03	0.664^{***}	0.689^{***}
		(0.184)	(0.181)
49	measurement_intercept_Envir04	0.185^{***}	0.198^{***}
		(0.0597)	(0.0586)
50	measurement_intercept_Envir05	0.128^{**}	0.144^{**}
		(0.0533)	(0.0579)
51	measurement_intercept_Envir06	0.277^{***}	0.315^{***}
		(0.0655)	(0.0737)
52	measurement_intercept_LifSty01	0.0742	0.0613
		(0.0475)	(0.0476)
53	measurement_intercept_LifSty07	-0.0521	-0.0608
		(0.0535)	(0.0547)
54	measurement_intercept_Mobil03	0.479^{***}	0.49^{***}
		(0.139)	(0.137)
55	measurement_intercept_Mobil05	0.786^{***}	0.829^{***}
		(0.266)	(0.264)
56	measurement_intercept_Mobil08	0.0853	0.0849
		(0.069)	(0.0768)
57	measurement_intercept_Mobil09	0.67^{***}	0.775^{***}
		(0.171)	(0.184)
58	measurement_intercept_Mobil10	1.22^{*}	1.35^{*}

			(0.637)	(0.814)
59	measurement_intercept_Mobil12		0.636	0.676
			(0.462)	(0.463)
60	measurement_intercept_NbCar		2.06***	2.04***
			(0.139)	(0.149)
61	struct_car_centric_attitude_ScaledIncome		0.00453	0.0171
			(0.0109)	(0.0108)
62	struct_car_centric_attitude_age_30_less		0.489***	0.437***
			(0.172)	(0.162)
63	struct_car_centric_attitude_car_oriented_parents		0.324***	0.312***
			(0.0998)	(0.0919)
64	struct_car_centric_attitude_employees		−0.0609	−0.0551
			(0.0813)	(0.0771)
65	struct_car_centric_attitude_high_education		−0.451***	−0.396***
			(0.106)	(0.099)
66	struct_car_centric_attitude_sigma_log		−0.17*	−0.228**
			(0.102)	(0.102)
67	struct_car_centric_attitude_top_manager		0.146	0.199
			(0.135)	(0.134)
68	struct_environmental_attitude_ScaledIncome		0.0195***	0.0228***
			(0.00577)	(0.00603)
69	struct_environmental_attitude_artisans		0.129**	0.136**
			(0.0588)	(0.0622)
70	struct_environmental_attitude_childSuburb		0.0773**	0.0857***
			(0.0306)	(0.0329)
71	struct_environmental_attitude_city_center_as_kid		0.151**	0.169**
			(0.0731)	(0.084)
72	struct_environmental_attitude_high_education		0.14***	0.161***
			(0.0461)	(0.0499)
73	struct_environmental_attitude_low_education		0.045	0.0512
			(0.0306)	(0.0339)
74	struct_environmental_attitude_sigma_log		−1.32***	−1.23***
			(0.232)	(0.213)
Number of observations		896	896	896

Number of parameters	7	60	75
Akaike Information Criterion	1039.0	34139.1	35014.7
Bayesian Information Criterion	1072.6	34427.0	35374.5

Standard errors: ***: $p < 0.01$, **: $p < 0.05$, *: $p < 0.1$

6 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.

7 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 12: Description of variables

8 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.

8.1 optima.py

```python
"""
.. _optima_data:

Data preparation
================

Optima data preparation for Biogeme.

This module reads the Optima case-study dataset from the tab-separated file
``optima.dat`` and prepares a :class:`biogeme.database.Database` object.

The preparation consists of:

- **Filtering / cleaning**: remove observations that are incompatible with the
  modeling assumptions (invalid choices, non-workers, tours with a single trip,
  and observations with zero travel times or distance).
- **Feature engineering**: create derived variables used later in the choice and
  latent-variable specifications (e.g., normalized weights, categories, scaled
  variables).
- **Convenience exports**: expose a large set of
  :class:`biogeme.expressions.Variable` objects corresponding to columns in the
  dataset so that other modules can simply import them from here.

Michel Bierlaire
2023-04-12
"""

import biogeme.database as db
import pandas as pd
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)
```

```python
    # 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')
```

8.2 config.py

```
"""

Configuration
=============

Central configuration for running estimation scripts.

This module defines the :class:`Config` dataclass, which gathers all high-level
options controlling the behavior of the estimation pipeline. A single instance
```

```
of :class:`Config` is typically created in a small configuration file (e.g.
`conf_01.py`, `conf_02.py`) and passed to the main execution routine.

The goal of this module is to provide a clear, typed, and immutable container
for experimental settings, so that the same codebase can be reused across
multiple configurations without duplication.

Michel Bierlaire
Thu Dec 25 2025, 08:08:37
"""

from dataclasses import dataclass
from typing import Literal


@dataclass(frozen=True)
class Config:
    """Configuration of a single estimation run.

    Each field controls a specific modeling or estimation choice:

    - ``name``: Human-readable identifier for the configuration (used for logging
      and output naming).
    - ``latent_variables``: Specifies whether latent variables are included in the
      model (``"zero"``) or whether the full hybrid choice model with two latent
      variables is used (``"two"``).
    - ``choice_model``: Indicates whether a discrete choice model is included
      alongside the latent-variable measurement model (``"yes"`` or ``"no"``).
    - ``estimation``: Estimation paradigm, either Bayesian (``"bayes"``) or
      maximum likelihood (``"ml"``).
    - ``number_of_bayesian_draws_per_chain``: Number of posterior draws per MCMC
      chain when Bayesian estimation is used.
    - ``number_of_monte_carlo_draws``: Number of Monte Carlo draws used for
      numerical integration in maximum likelihood estimation.

    The dataclass is frozen to guarantee immutability during execution and
    improve reproducibility.
    """

    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).

8.3 latent_variables.py

```
"""

Latent variables
================

Definitions of latent variables used in the hybrid choice model.

This module centralizes the specification of latent variables that enter the
hybrid choice (MIMIC) model. For each latent variable, it defines:

- the name of the latent variable,
- the list of explanatory variables entering its structural equation, and
- the set of Likert-type indicators used in its measurement equations.

The goal is to keep all latent-variable metadata in a single, transparent
location, making the model specification easier to read, maintain, and
modify.

The variables defined here are imported by higher-level model construction
code and should therefore remain lightweight and declarative (no model logic
is implemented in this file).

Michel Bierlaire
```

```
Thu Dec 25 2025, 08:13:19
"""

"""Latent variable representing the car-centric attitude.

This latent variable captures preferences and attitudes related to car
ownership and car-oriented lifestyles. It is explained by socio-demographic
and background variables and measured using a set of mobility, lifestyle,
and environment-related Likert indicators.
"""
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 representing the environmental attitude.

This latent variable captures environmental awareness and sensitivity. Its
structural equation depends on socio-demographic and residential background
variables, and it is measured using a set of environment-, mobility-, and
lifestyle-related Likert indicators.
"""
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',
}
```

8.4 likert_indicators .py

```
"""

Likert indicators
=================

Definition of Likert indicators and Likert scale types used in the hybrid choice model.

This module centralizes the specification of all Likert-type survey indicators
used in the measurement equations of the hybrid choice (MIMIC) model, together
with the definition of the corresponding Likert scale types.

It contains two main objects:

- ``likert_indicators``: a list of :class:`LikertIndicator` objects, each
  corresponding to a survey question (statement) and its associated metadata.
- ``likert_types``: a list of :class:`LikertType` objects, defining how different
  types of indicators are mapped to ordered categories and threshold
  parameterizations.

```

```
The content of this file is purely declarative: no estimation or model logic is
implemented here. The definitions are imported by higher-level model-building
code.

Michel Bierlaire
Thu Dec 25 2025, 08:14:12
"""

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

"""List of Likert indicators used in the model.

Each entry corresponds to one survey item and specifies:

- a unique indicator name,
- the text of the statement presented to respondents,
- the indicator type, which determines the associated Likert scale
  specification defined in ``likert_types``.

The indicators cover environmental attitudes, mobility-related perceptions,
lifestyle preferences, and the number of cars in the household.
"""
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',
    ),
]

"""Definition of Likert scale types and their threshold structures.

Each :class:`LikertType` defines how responses for a given indicator type are
modeled, including:

- whether the thresholds are symmetric or non-symmetric,
- the ordered response categories,
- labels corresponding to neutral or missing responses,
- normalization rules for the scale parameters.

These specifications are shared across all indicators of the same type.
"""
likert_types = [
    LikertType(
        type='likert',
        symmetric=True,
        categories=[1, 2, 3, 4, 5],
        neutral_labels=[6, -1],
        scale_normalization='Envir01',
    ),
    LikertType(
        type='cars',
        symmetric=False,
        categories=[0, 1, 2, 3],
        neutral_labels=[-1],
        fix_first_cut_point_for_non_symmetric_thresholds=0.0,
        scale_normalization='NbCar',
    ),
]
```

8.5 mimic.py

```
"""

MIMIC model
===========

Construction of the MIMIC (latent-variable) component of the hybrid choice model.

This module defines a helper function that builds and returns an
:class:`OrderedMimic` object, fully configured according to a given
:class:`Config`.

The MIMIC model includes two latent variables:

- a car-centric attitude latent variable, and
- an environmental attitude latent variable.

For each latent variable, the module specifies:

- the structural equation (explanatory variables),
- the associated Likert-type indicators, and
- the normalization used for identification.

The resulting MIMIC model is used by higher-level code to construct
measurement equations and to combine them with the choice model, under
either maximum likelihood or Bayesian estimation.

```

```python
This file is intentionally limited to model specification; it contains no
data handling or estimation logic.

Michel Bierlaire
Thu Dec 25 2025, 08:15:26
"""

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):
    """Generate and return a configured MIMIC model.

    The estimation mode (Bayesian or maximum likelihood) is selected based on
    ``config.estimation``. The function then:

    1. Creates an :class:`OrderedMimic` container with the appropriate
       estimation mode and Likert indicator definitions.
    2. Defines the car-centric and environmental latent variables, including
       their structural equations, indicators, and normalizations.
    3. Registers both latent variables with the MIMIC model.

    :param config: Global configuration object controlling the estimation
        paradigm.
    :return: A fully specified :class:`OrderedMimic` instance.
    """
    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.

8.6 choice_model.py

```python
"""

Choice model
============

Choice model specification with optional latent-variable interactions.

This module defines the utility functions for a discrete choice model that can
optionally incorporate latent variables estimated in a hybrid choice (MIMIC)
framework.

The behavior of the choice model is controlled by the `Config` object, in
particular by the attribute `config.latent_variables`, which can take two
values:

- ``"zero"``: no latent variables are used; the choice model reduces to a
  standard discrete choice model with observed attributes only.
- ``"two"``: two latent variables (car-centric and environmental attitudes)
  enter the utilities through interaction terms.

Latent variables affect the model in two ways:

1. Multiplicative effects on time coefficients, where the base time sensitivity
   is scaled by an exponential function of the latent variable(s).
2. Additive alternative-specific constants that shift utilities depending on
   the latent attitudes.

This module only constructs the systematic utility functions. The likelihood
construction (logit, Monte Carlo integration, Bayesian or maximum-likelihood
wrapping) is handled elsewhere in the codebase.

Michel Bierlaire
Thu Dec 25 2025, 08:07:33
"""

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 == 'two'``: both car-centric and environmental latent
        variables enter.

    Note: this function returns only the utilities. The estimation / likelihood wrapping
    is handled elsewhere.
    """
    if config.latent_variables not in {"zero", "two"}:
        raise ValueError(
            f"config.latent_variables must be 'zero' or 'two', got
                {config.latent_variables!r}."
        )
    include_latent_variables = config.latent_variables == "two"

    # Latent variables can be: zero (none) 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)
```

```python
    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.

8.7 estimate.py

```python
"""

Model estimation
================

Estimation pipeline for discrete and hybrid choice models.

This module defines the high-level logic used to construct and estimate
choice models and hybrid choice (MIMIC) models using Biogeme. The behavior
of the pipeline is fully controlled by a :class:`Config` object, allowing
the same code to be reused across multiple experimental configurations.

Depending on the configuration, the module can:

- estimate a standard discrete choice model without latent variables,
- estimate a hybrid choice model with two latent variables and measurement
  equations,
- perform either maximum likelihood or Bayesian estimation,
- optionally combine the choice model likelihood with the measurement
  likelihood.

The module is intentionally declarative: model structure is assembled here,
while data handling, estimation, and result post-processing are delegated
```

```python
to specialized helper modules.

Michel Bierlaire
Thu Dec 25 2025, 08:13:25
"""

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 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:
    """Generate the likelihood expression to be estimated.

    This function constructs the Biogeme expression corresponding to the
    selected model configuration.

    - If ``config.latent_variables == 'zero'``, only the discrete choice
      model likelihood is returned.
    - Otherwise, the likelihood from the latent-variable measurement
      equations is constructed and optionally combined with the choice
      model likelihood.

    The expression is wrapped differently depending on the estimation
    paradigm:

    - Bayesian estimation uses log-likelihood expressions (``loglogit`` and
      log measurement equations).
    - Maximum likelihood estimation uses Monte Carlo integration when
      required.

    :param config: Configuration object controlling model structure and
        estimation mode.
    :return: A Biogeme expression representing the full likelihood to be
        estimated.
    """
    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)

    # 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:
    """Estimate the model specified by the given configuration.

    This function:

```

```python
    1. Builds the likelihood expression using :func:`generate_expression`.
    2. Creates and configures a :class:`BIOGEME` object.
    3. Either reads existing estimation results from disk or runs a new
       estimation.
    4. Prints a short textual summary and displays the estimated parameters
       in a pandas table.

    :param config: Configuration object defining the model specification,
        estimation method, and numerical settings.
    """
    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)
```

8.8 read_or_estimate.py

```python
"""
Read or estimate model parameters
=================================

Utility functions to either **read previously estimated parameters from disk**
or **run a new estimation** if no results are available.

This module provides a lightweight abstraction around Biogeme's estimation
routines, allowing scripts to be written in a reproducible way without
manually checking whether estimation results already exist.

Both maximum likelihood and Bayesian estimation paradigms are supported.

Michel Bierlaire (EPFL)
Thu Dec 25 2025, 08:28:26
"""

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:
    """Read estimation results from disk or estimate the model if needed.

    Depending on the selected estimation mode, this function attempts to read
    existing results from disk:

    - Bayesian estimation: results are read from a NetCDF file (``.nc``).
    - Maximum likelihood estimation: results are read from a YAML file
      (``.yaml``).

    If the corresponding file is not found, the model is estimated and the
    results are returned.

    This mechanism ensures that expensive estimations are not rerun
    unnecessarily while keeping the calling code simple and declarative.

    :param the_biogeme: Configured :class:`biogeme.biogeme.BIOGEME` object.
    :param estimation_mode: Estimation mode, either
        :class:`EstimationMode.BAYESIAN` or
        :class:`EstimationMode.MAXIMUM_LIKELIHOOD`.
    :param directory: Directory where result files are expected to be found.
    :return: Estimation results, either
        :class:`EstimationResults` or :class:`BayesianResults`.
    :raises ValueError: If an unsupported estimation mode is provided.
    """
    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
```

8.9 plot_b01_choice_only_ml.py

```
"""
1. Choice model only - maximum likelihood estimation
====================================================

This script runs a **standard discrete choice model** without any latent
variables, estimated by **maximum likelihood** using Biogeme.

It serves as a baseline specification against which hybrid choice models
(with latent variables and measurement equations) can be compared.

The configuration is defined locally in this file and passed to the generic
estimation pipeline via :func:`estimate_model`.

Michel Bierlaire
Thu Dec 25 2025, 08:24:06
"""

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)
```

8.10 plot_b02_mimic_ml.py

```
"""
2. MIMIC model - maximum likelihood estimation
=============================================

This script estimates a **pure MIMIC model** (measurement and structural
components only) using **maximum likelihood**, without an associated discrete
choice model.

It is mainly intended to:

- test and validate the latent-variable specification,
- assess identification and normalization issues, and
- serve as a building block for hybrid choice models.

The model configuration is defined locally in this file and passed to the
generic estimation pipeline via :func:`estimate_model`.

Michel Bierlaire
Thu Dec 25 2025, 08:24:35
"""

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)
```

8.11 plot_b03_hybrid_ml.py

```
"""
3. Hybrid choice model - maximum likelihood estimation
======================================================

This script estimates a **hybrid choice model** that combines:

- a discrete choice model, and
- a MIMIC model with two latent variables (structural and measurement equations),

using **maximum likelihood estimation** in Biogeme.

It represents the full model specification, bringing together the choice
component and the latent-variable component, and can be compared against:

- the choice-only model, and
- the MIMIC-only model,

to assess the contribution of latent variables to model performance.

The configuration is defined locally in this file and passed to the generic
estimation pipeline via :func:`estimate_model`.

Michel Bierlaire
Thu Dec 25 2025, 08:25:28
"""

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

estimate_model(config=the_config)
```

8.12 plot_b04_choice_only_bayes.py

```
"""
4. Choice model only - Bayesian estimation
=========================================

This script estimates a **standard discrete choice model** without any latent
variables using **Bayesian estimation** in Biogeme.

It serves as the Bayesian counterpart of the choice-only maximum likelihood
specification and provides a baseline for comparison with:

- the Bayesian hybrid choice model, and
- the corresponding maximum likelihood estimates.

The configuration is defined locally in this file and passed to the generic
estimation pipeline via :func:`estimate_model`.

Michel Bierlaire
Thu Dec 25 2025, 08:27:04
"""

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)
```

8.13 plot_b05_mimic_bayes.py

```
"""
5. MIMIC model - Bayesian estimation
===================================

This script estimates a **pure MIMIC model** (latent-variable structural and
measurement equations only) using **Bayesian estimation** in Biogeme, without
an associated discrete choice model.

It is primarily intended to:

- assess identification and normalization under Bayesian inference,
- inspect posterior distributions of latent-variable parameters, and
- provide a Bayesian benchmark for comparison with the maximum likelihood
  MIMIC specification.

The configuration is defined locally in this file and passed to the generic
estimation pipeline via :func:`estimate_model`.

Michel Bierlaire
Thu Dec 25 2025, 08:27:04
"""

```

```
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)
```

8.14 plot_b06_hybrid_bayes.py

```
"""

6. Hybrid choice model - Bayesian estimation
============================================

This script estimates the **full hybrid choice model**, combining:

- a discrete choice model, and
- a MIMIC model with two latent variables (structural and measurement equations),

using **Bayesian estimation** in Biogeme.

It represents the most complete specification in the model family and is
primarily used to:

- study identification and normalization under Bayesian inference,
- analyze posterior distributions of both choice and latent-variable parameters,
- compare Bayesian and maximum likelihood hybrid models, and
- assess the added value of latent variables relative to simpler specifications.

The configuration is defined locally in this file and passed to the generic
estimation pipeline via :func:`estimate_model`.

Michel Bierlaire
Thu Dec 25 2025, 08:27:43
"""

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.