Estimating hybrid choice models with Biogeme

Michel Bierlaire Moshe Ben-Akiva Joan Walker

August 14, 2026

Report TRANSP-OR 260814
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 3.3.3 and 3.3.4.

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, Walker, Bernardino, Gopinath, Morikawa and Polydoropoulou, 2002, Ben-Akiva et al., 2002, to cite just a few early references). This document presents a practical introduction to the specification and estimation of hybrid choice models using Biogeme. It introduces the main concepts and notation underlying these models, discusses the associated normalization issues, and illustrates their implementation through a sequence of progressively richer examples. The examples show how latent variables, structural equations, measurement equations, and choice models can be combined within a unified estimation framework.

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. An asterisk (*) is used to denote latent variables, distinguishing them from observed variables. 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, and assuming conditional independence of the indicators given x^*_n, is given by:

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

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.

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

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

where $f(x^*_n | x_n; \alpha)$ is the conditional joint probability density function of the latent variables x^*_n given the observed explanatory variables x_n. The vector α contains all parameters appearing in the structural equations (2), including the coefficients ψ and the scale parameters of the structural disturbances. 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 the case study, valid responses are represented by the ordered labels $1, 2, 3, 4, 5$. The numerical labels are arbitrary; only their ordering matters. The values 6 and -1 are non-informative labels and are not valid response categories.

In practice, ordinal indicators are sometimes treated as if they were continuous variables. This approximation is common when the scale contains several ordered categories and the distances between adjacent categories are assumed to be approximately equal. In that case, the indicator can be modeled using the continuous measurement equation presented in the previous section. If, however, the discrete and ordered nature of the indicator is to be explicitly taken into account, a different specification is required. The approach presented below models the indicator as an ordered discrete variable and directly accounts for the ordinal structure of the response scale.

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:

$$I_{n\ell} = \begin{cases} j_1 & \text{if } I^*_{n\ell} < \tau_{\ell 1}, \\ j_2 & \text{if } \tau_{\ell 1} \leqslant I^*_{n\ell} < \tau_{\ell 2}, \\ \vdots & \\ j_m & \text{if } \tau_{\ell, m-1} \leqslant I^*_{n\ell} < \tau_{\ell m}, \\ \vdots & \\ j_M & \text{if } \tau_{\ell, M_\ell - 1} \leqslant I^*_{n\ell}, \end{cases} \tag{8}$$

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

$$\tau_{\ell 1} \leqslant \tau_{\ell 2} \leqslant \cdots \leqslant \tau_{\ell M_\ell - 1}. \tag{9}$$

The formulation above associates a distinct set of threshold parameters $\tau_{\ell 1}, \dots, \tau_{\ell, M_\ell - 1}$ with each indicator ℓ. In practice, this specification may involve a large number of parameters when many ordinal indicators are included in the model. A common simplification is to impose a common set of thresholds for groups of indicators that share the same response scale and are intended to measure similar constructs. In that case, the thresholds are no longer indicator-specific and the index ℓ can be dropped.

From a statistical perspective, this restriction reduces the flexibility of the model, as the relative distances between response categories are assumed to be identical across the corresponding indicators. The model is therefore more constrained than the fully general specification. However, the resulting reduction in the number of parameters can be substantial, leading to more stable estimation, shorter computation times, and improved interpretability. Whether such a simplification is appropriate depends on the application and on the degree of similarity among the indicators.

The case study presented in Section 4 adopts this simplified specification and uses a common set of thresholds across all ordinal indicators.

Defining $\tau_{\ell 0} = -\infty$ and $\tau_{\ell M_\ell} = +\infty$, it simplifies to

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

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 probability of observing category j_m is equal to the probability that the latent response $I^*_{n\ell}$ falls between the two thresholds defining that category, namely $\tau_{\ell, m-1}$ and $\tau_{\ell, m}$. Therefore, the contribution of indicator ℓ for observation n to the likelihood function, conditional on the latent variables, can be derived directly from the distribution of the disturbance term:

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

where j_m is the observed category for respondent n and indicator ℓ, and $\Phi(\cdot)$ denotes the cumulative distribution function of the standard normal distribution.

The probability of observing a category is therefore obtained as the difference between two cumulative probabilities. Intuitively, the latent response must fall within the interval delimited by the two thresholds associated with that category. The above specification computes this probability using the normal distribution of the disturbance term. It is known as the *ordered probit model* and is widely used for modeling ordinal responses that depend on latent constructs.

An alternative specification is obtained by assuming that the disturbance term follows a standard logistic distribution,

$$\upsilon_{n\ell} \sim \text{Logistic}(0, 1),$$

instead of a standard normal distribution. In that case, the cumulative distribution function $\Phi(\cdot)$ in Eq. (12) is replaced by the logistic cumulative distribution function

$$F(t) = \frac{1}{1 + \exp(-t)},$$

yielding the *ordered logit model*. The resulting category probabilities are therefore

$$\Pr(I_{n\ell} = j_m \mid x^*_n, x_n) = F\left(\frac{\tau_{\ell,m} - \lambda_{\ell 0} - \sum_k \lambda_{\ell k} x^*_{nk}}{\sigma_{\upsilon\ell}}\right) - F\left(\frac{\tau_{\ell,m-1} - \lambda_{\ell 0} - \sum_k \lambda_{\ell k} x^*_{nk}}{\sigma_{\upsilon\ell}}\right). \tag{13}$$

The ordered probit and ordered logit models differ only through the assumed distribution of the disturbance term. Since the normal and logistic cumulative distribution functions have similar shapes, the two specifications often lead to very similar empirical results. Both models are implemented in Biogeme and illustrated in the examples of Section 4.

If we observe a vector of ordinal 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:

$$\prod_{\ell=1}^{L_n} \Pr(I_{n\ell} = j_m | x^*_n, x_n; \lambda_\ell, \Sigma_{\upsilon\ell}). \tag{14}$$

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

$$\int_{x^*_n} \left[\prod_{\ell=1}^{L_n} \Pr(I_{n\ell} = j_m | x^*_n, x_n; \lambda_\ell, \Sigma_{\upsilon\ell})\right] f(x^*_n | x_n; \alpha)\, dx^*_n, \tag{15}$$

where $f(x_n^*|x_n; \alpha)$ is the conditional pdf of the latent variables given the observed explanatory variables, and α denotes the parameters of the structural equations. It includes the coefficients ψ of the structural equations as well as the parameters governing the distribution of the structural disturbances in Equation (2). Again, this integral is approximated using Monte-Carlo integration.

2 Normalization and identification

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 between normalizations that identify the *latent variables themselves* and those that identify the *ordinal measurement layers*. In principle, there are infinitely many equivalent normalization schemes, all leading to statistically equivalent models and identical behavioral conclusions. The choice of a normalization is therefore a matter of convention rather than substance. Among the possible approaches, we present the reference-indicator strategy, which is widely used because of its interpretability.

2.1 Why normalization is required

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

At the level of latent variables, two fundamental sources of non-identification arise:

- **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.

The need for normalization is closely analogous to identification issues in logit and related random utility models. Utilities are latent and only differences in utility matter; as a consequence, utilities are invariant to the addition of a constant and to multiplication by a positive scalar. Standard practice therefore requires fixing one alternative-specific constant (location) and fixing the utility scale, typically by normalizing the variance of the error term.

Latent variables play a similar role: they enter the likelihood only through systematic components and relative comparisons. Measurement equations — whether continuous or ordinal — therefore require explicit normalizations to fix both location and scale.

2.2 Normalization of latent variables

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

For each latent variable, exactly two normalizations are therefore required:

- one to fix its *location*;

- one to fix its *scale*.

These two normalizations are necessary and sufficient to remove the two degrees of freedom associated with location and scale invariance. Any additional restriction affecting these dimensions would be redundant and may lead to over-constrained specifications.

A practical and interpretable approach is the *reference-indicator strategy.* Consider the measurement equation (4). For each latent variable k, one indicator $\ell(k)$ is selected as its reference indicator. Location is fixed by setting the corresponding intercept parameter $\lambda_{\ell(k)0}$ to zero, and scale is fixed by setting the corresponding factor loading $\lambda_{\ell(k)k}$ to either $+1$ or -1. The sign choice 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 and non-reference loadings—become meaningful and interpretable.

2.3 Normalization of ordinal measurement models

We now consider ordinal indicators modeled using ordered response specifications, such as the ordered probit and ordered logit models introduced in Section 1.3. In this case, the observed response reveals only the interval in which an underlying latent response lies. This introduces additional sources of non-identification beyond those associated with the latent variables themselves.

For each ordered probit measurement equation, two additional invariances arise:

- **Threshold-location invariance**: adding the same constant to the latent response, the measurement intercept, and all thresholds leaves observed probabilities unchanged;

- **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 independent set of thresholds*, that is, to each group of indicators that shares a common latent response scale and threshold parameters. Consequently, normalization of the ordinal measurement layer must be performed *separately for each threshold system*, even when multiple indicators share the same response scale.

Identification therefore requires, for each threshold system, one normalization to fix *location* and one to fix *scale*.

The phrase *threshold system* is important here. If each ordinal indicator has its own thresholds, then each indicator defines its own threshold system and requires its own location and scale normalizations. If, instead, several indicators are constrained to share the same thresholds, these indicators form a single threshold system. In that case, the location and scale normalizations are imposed only once for the group, not once per indicator.

For example, if a set of Likert indicators is modeled with common thresholds

$$\tau_1, \ldots, \tau_{M-1},$$

then one location normalization, such as fixing one threshold or imposing symmetry around zero, and one scale normalization, such as fixing the ordered-probit error variance, identify the ordinal measurement layer for the whole group. Imposing the same normalizations separately for each indicator in the group would be redundant, because the indicators no longer have separate threshold locations or scales.

Using common thresholds therefore reduces not only the number of threshold parameters, but also the number of ordinal-layer normalizations required. The general rule is that normalizations are attached to independent threshold systems, not to individual indicators.

Location normalization. The location of the ordinal measurement layer is not identified. One location normalization is therefore required for each independent threshold system.

A common approach is to fix one threshold, for example

$$\tau_{\ell 1} = 0,$$

which anchors the location of the latent response scale.

An alternative is to impose a centered threshold structure. When the response categories are themselves designed to be symmetric, as in many Likert scales (e.g., strongly disagree, disagree, neutral, agree, strongly agree), it may be reasonable to constrain the threshold system to be symmetric around zero. This specification simultaneously fixes the location of the threshold system and imposes an additional structural assumption reflecting the symmetry of the response scale. It also reduces the number of free parameters and often improves interpretability.

When symmetry is imposed, a convenient parameterization for $M_\ell = 5$ response categories is:

$$\begin{aligned}
\tau_{\ell 1} &= -\delta_{\ell 0} - \delta_{\ell 1}, \\
\tau_{\ell 2} &= -\delta_{\ell 0}, \\
\tau_{\ell 3} &= \delta_{\ell 0}, \\
\tau_{\ell 4} &= \delta_{\ell 0} + \delta_{\ell 1},
\end{aligned}$$

with $\delta_{\ell 0} > 0$ and $\delta_{\ell 1} > 0$. This guarantees strict ordering and centers the threshold system at zero, aligning the statistical specification with the semantics of balanced response scales.

Scale normalization. Ordered response models do not identify the scale of the latent response relative to threshold spacing and measurement noise. A scale normalization is therefore required for each independent threshold system.

For both ordered probit and ordered logit models, the disturbance term is assumed to follow a standardized distribution (respectively a standard normal or a standard logistic distribution). The scale of the latent response is then governed by the measurement scale parameter $\sigma_{\upsilon\ell}$. Because only relative scales are identified, one scale normalization must be imposed for each independent threshold system.

When several indicators share a common threshold system, this normalization is applied only once for the entire group. When different groups of indicators have distinct threshold systems, a separate scale normalization is required for each group.

2.4 Summary

The normalizations discussed above can be understood as fixing the *origins* and *units* of unobserved scales. Reference-indicator restrictions anchor each latent variable to an observed indicator, thereby fixing its location and scale. Separately, each ordered probit threshold system requires its own location and scale normalization, because only differences between the latent response and thresholds are identified.

Table 1 summarizes these requirements and makes explicit how often each restriction must be applied to avoid redundancy.

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 recommended 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)	Location invariance	Fix one threshold to zero *or* impose symmetry	Once per independent threshold system
Ordinal measurement (ordered probit)	Scale invariance	Fix measurement error scale (e.g. $\sigma = 1$)	Once per independent 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)).

More specifically, the latent variables are generated by the structural equations (1) (or their linear specification (2)), and they enter the measurement equations (3) or (11), depending on whether the indicators are continuous or ordinal.

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.

As in the likelihood expressions derived in Section 1, we assume that the indicators are conditionally independent given the latent variables. 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:

$$\Lambda_n^I(I_n \mid x_n^*, x_n; \lambda, \sigma_\upsilon, \tau) = \prod_{\ell=1}^{L_n} g_\ell(I_{n\ell} \mid x_n^*, x_n; \lambda_\ell, \sigma_{\upsilon\ell}, \tau_\ell), \tag{16}$$

where

$$g_\ell(\cdot) = \begin{cases} \text{normal density implied by Eq. (4),} \\ \text{ordered-probit probability given by Eq. (12).} \end{cases}$$

For continuous indicators, g_ℓ depends on the measurement parameters λ_ℓ and the scale parameter $\sigma_{\upsilon\ell}$. For ordinal indicators, it also depends on the threshold parameters τ_ℓ, or on the common threshold system τ when thresholds are shared across indicators.

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:

$$\Lambda_n^{\text{MIMIC}}(\alpha, \lambda, \sigma_\upsilon, \tau) = \int_{x_n^*} \Lambda_n^I(I_n \mid x_n^*, x_n; \lambda, \sigma_\upsilon, \tau) f(x_n^* \mid x_n; \alpha) dx_n^*, \tag{17}$$

where $f(x_n^*|x_n; \alpha)$ denotes the joint density of the latent variables implied by the structural equations (1)–(2).

This expression is the direct extension of the unconditional likelihoods derived in Eqs. (6) and the corresponding ordinal case.

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.

In the notation introduced in Section 1, the latent variables x^*_n generated by the structural equations become additional explanatory variables in the utility specification.

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

$$P(i_n \mid x^*_n, x_n; \beta, \mu) \;=\; \frac{\exp(\mu\, V(x_{in}, x^*_n; \beta))}{\sum_{j \in \mathcal{C}_n} \exp(\mu\, V(x_{jn}, x^*_n; \beta))}, \tag{18}$$

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 observed data consist of the choice i_n and the vector of indicators I_n. Conditional on x^*_n, the hybrid choice model contribution to the likelihood is the product of:

- the conditional choice probability from Eq. (18),
- the conditional indicator likelihood from Eq. (16).

That is,

$$\Lambda_n(i_n, I_n \mid x^*_n, x_n; \beta, \mu, \lambda, \sigma_\upsilon, \tau) = P(i_n \mid x^*_n, x_n; \beta, \mu)\; \Lambda^I_n(I_n \mid x^*_n, x_n; \lambda, \sigma_\upsilon, \tau)\,. \tag{19}$$

This factorization relies on the assumption that, conditional on the latent variables, the choice and the indicators are independent.

Because x^*_n is latent, the individual likelihood contribution integrates (19) over $f(x^*_n \mid x_n; \alpha)$ implied by the structural equations (2):

$$\Lambda^{\text{HCM}}_n(\alpha, \beta, \mu, \lambda, \sigma_\upsilon, \tau) = \int_{x^*_n} P(i_n \mid x^*_n, x_n; \beta, \mu)\Lambda^I_n(I_n \mid x^*_n, x_n; \lambda, \sigma_\upsilon, \tau) f(x^*_n \mid x_n; \alpha)\, dx^*_n. \tag{20}$$

Let θ denote the full parameter vector, including the utility parameters β, the scale parameter μ, the structural parameters α, the measurement parameters λ, the threshold parameters τ for ordinal indicators, and any scale parameters. The parameter vector θ is assumed to satisfy the normalization conditions discussed in Section 2.

The sample log-likelihood is

$$\begin{aligned} \mathcal{L}(\theta) &= \sum_n \ln \Lambda^{\text{HCM}}_n(\alpha, \beta, \mu, \lambda, \sigma_\upsilon, \tau) \\ &= \sum_n \ln \left[\int_{x^*_n} P(i_n \mid x^*_n, x_n; \beta, \mu)\Lambda^I_n(I_n \mid x^*_n, x_n; \lambda, \sigma_\upsilon, \tau) f(x^*_n \mid x_n; \alpha)\, dx^*_n \right]. \end{aligned} \tag{21}$$

The integral in (21) typically has no closed form and is evaluated by Monte-Carlo integration.

4 A case study

This case study illustrates, step by step, how the modeling ingredients introduced in Sections 1, 2, and 3 can be implemented with Biogeme. Rather than presenting directly the most general hybrid specification, we adopt a progressive approach. At each step, only one modeling dimension is modified, so that the role of each component can be understood clearly: the choice model, the latent variables, the measurement equations, the estimation strategy, and the implementation style.

The empirical application is based on revealed-preference data collected in Switzerland in the context of the OPTIMA project (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 objective was to better understand travel behavior in low-density areas, which represent the typical service environment of CarPostal. In addition to observed mode choices and explanatory variables, the survey contains several psychometric indicators that can be used to infer latent constructs. The data file and its description are available on the Biogeme webpage. A description of the variables is also provided in Appendix A.

The sequence of examples is organized as a set of increasingly rich specifications. We start with a standard discrete choice model without latent variables, which serves as a reference specification. We then introduce latent variables in isolation through a MIMIC model, using the continuous case for the measurement equations in order to focus on the structural equation, the measurement equation, and the associated normalization issues. Once this foundation is established, the latent variable is incorporated into a mode choice model, first using a sequential estimation strategy and then using simultaneous estimation. This allows us to contrast a simplified estimation approach with the full hybrid formulation described in Section 3.

In a second stage, we replace the continuous measurement equations by ordinal measurement equations. This is an important step because the psychometric indicators available in the data are naturally interpreted as ordinal responses rather than as continuous measurements. Two variants are considered: ordered logit and ordered probit. The goal is not to change the behavioral content of the model, but to show how the same latent-variable and choice structure can be combined with different ordinal measurement schemes and with the corresponding normalization of threshold systems.

We now present the examples in the order described above.

4.1 Running the examples

The example files are located in `docs/source/examples/hybrid_choice_models`. The data file `optima.dat` is included in that directory. Because the scripts use local imports and read the data file using a relative path, they should be executed from this directory:

```
cd docs/source/examples/hybrid_choice_models
uv run python plot_h01_mode_logit.py
```

The recommended execution order is:

1. `plot_h01_mode_logit.py`: baseline mode-choice model;
2. `plot_h02_lv_mimic_gauss.py`: Gaussian MIMIC model;
3. `plot_h03_mode_lv_gauss_seq.py`: sequential hybrid model;
4. `plot_h04_mode_lv_gauss_simult.py`: simultaneous Gaussian model;
5. `plot_h05_mode_lv_ordlogit_simult.py`: simultaneous ordered-logit model;
6. `plot_h06_mode_lv_ordprobit_simult.py`: simultaneous ordered-probit model;
7. `plot_h07_mode_lv_ordprobit_simult_generated.py`: explicit generated implementation of H06.

The sequential example requires the YAML results produced by the MIMIC example. The last four examples are otherwise independent, although they all use the same data preparation and latent-variable structure.

The hybrid examples use 50,000 simulation draws and may require substantial computation time. For an initial smoke test, H04 can be run with fewer draws and fewer iterations as described below. Results created with a small number of draws should not be used for scientific interpretation.

4.2 Example 1: baseline mode choice model

We start with the simplest specification considered in this tutorial, a standard logit model for mode choice, without explicit latent variables. Its role is to provide a reference specification against which the richer models introduced later can be compared.

For each individual n, let $\mathcal{C}_n$ denote the set of available transport modes, and let $i_n \in \mathcal{C}_n$ be the observed mode choice. In the application, three alternatives are considered:

- public transport,
- car,
- slow modes.

Their systematic utilities depend only on observed explanatory variables. Using the notation introduced in Section 3.2, they are specified as follows.

For public transport,

$$V_{\mathrm{PT},n} = \beta_{\mathrm{asc,PT}} + \beta_{\mathrm{time,PT}}\,\mathrm{TimePT}_n + \beta_{\mathrm{cost}}\,\mathrm{MarginalCostPT}_n. \tag{22}$$

For car,

$$V_{\mathrm{Car},n} = \beta_{\mathrm{asc,Car}} + \beta_{\mathrm{time,Car}}\,\mathrm{TimeCar}_n + \beta_{\mathrm{cost}}\,\mathrm{CostCar}_n. \tag{23}$$

For slow modes,

$$V_{\mathrm{Slow},n} = \beta_{\mathrm{dist},n}\,\mathrm{Distance}_n, \tag{24}$$

where the distance coefficient is specified as a function of the trip purpose:

$$\beta_{\mathrm{dist},n} = \beta_{\mathrm{dist,work}}\,\mathbf{1}\{\mathrm{PurpHWH}_n = 1\} + \beta_{\mathrm{dist,other}}\,\mathbf{1}\{\mathrm{PurpHWH}_n \neq 1\}. \tag{25}$$

In these expressions, TimePT_n, $\mathrm{TimeCar}_n$, $\mathrm{MarginalCostPT}_n$, $\mathrm{CostCar}_n$, and $\mathrm{Distance}_n$ denote the observed travel attributes for individual n, while $\mathbf{1}\{\cdot\}$ is the indicator function selecting work trips from all other trip purposes.

As discussed in Section 2, discrete choice models require normalization to ensure identification. In the present specification, the location of the utility scale is fixed by setting the alternative-specific constant associated with slow modes to zero. This normalization is implicit in the specification (24), where no constant term is included for slow modes.

The scale of utility is identified by normalizing the cost coefficient to

$$\beta_{\mathrm{cost}} = -1.$$

Indeed, only relative utility differences are identified in discrete choice models, and multiplying all utility coefficients by a common positive constant leaves the choice probabilities unchanged up to a corresponding rescaling of the logit scale parameter. Fixing the cost coefficient therefore defines the unit in which utility is measured.

With this normalization in place, the scale parameter appearing in the logit model can be estimated. The systematic utilities are multiplied by a positive scale parameter μ.

At this stage, the systematic utilities depend only on observed variables. No additional latent construct x_n^* is introduced in the specification. The objective of this first example is therefore to isolate the choice component before augmenting it later with structural equations for latent variables and with psychometric indicators.

Under the logit specification introduced in Eq. (18), the probability of the observed choice is

$$P(i_n \mid x_n; \beta) = \frac{\exp(\mu V(x_{i_n,n}; \beta))}{\sum_{j \in \mathcal{C}_n} \exp(\mu V(x_{jn}; \beta))}, \tag{26}$$

where μ is the scale parameter of the logit model and β is the vector collecting all utility parameters.

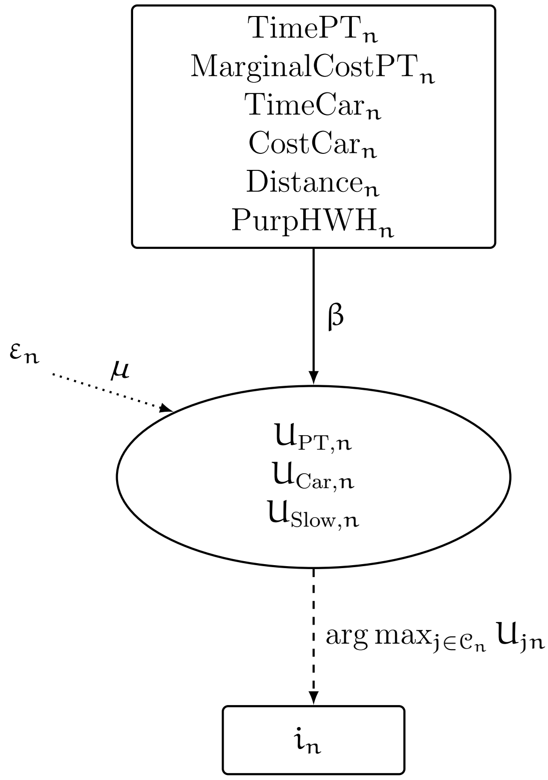

Figure 1: Graphical representation of the baseline mode choice model.

Figure 1 provides a graphical representation of the baseline mode choice model.

The observed explanatory variables x_n, listed in the upper node, enter the specification of the random utility functions $U_{\text{PT},n}$, $U_{\text{Car},n}$, and $U_{\text{Slow},n}$ through their deterministic component. This corresponds to the structural specification of the model, where the systematic part of utility is constructed as a function of observed attributes and parameters β.

The random term ε_n captures unobserved factors affecting utility. It enters the utility functions with a scale parameter μ, reflecting the normalization of the variance of the error terms.

The lower node represents the observed choice i_n, which is linked to the latent utilities through the utility-maximization rule. The dashed arrow captures this measurement relation: the observed choice reveals which alternative attains the maximum utility. Under the assumption that the error terms are i.i.d. extreme value, this behavioral rule leads to the logit probability model defined in Eq. (18).

Because no additional latent variable x_n^* is introduced in this first example, there is no structural equation of the form (2) describing latent constructs beyond the utility functions themselves, and no psychometric indicators contributing through the conditional indicator likelihood (16). As a consequence, the individual contribution to the likelihood reduces to the choice component alone:

$$\Lambda_n(\beta) = P(i_n \mid x_n; \beta). \tag{27}$$

The sample log-likelihood is therefore

$$\mathcal{L}(\beta) = \sum_n \ln P(i_n \mid x_n; \beta). \tag{28}$$

This specification is implemented in the script `plot_h01_mode_logit.py`. The corresponding utility functions are generated in the module

`choice_latent_variables.py`.

Although that module is written so that latent-variable terms can be added later, these terms are not activated in the present example.

This first example is deliberately simple. It establishes the choice component of the case study before introducing, in the following examples, explicit latent variables, structural equations, indicator equations, normalization issues, and the Monte-Carlo integration required to evaluate the hybrid choice model likelihood (21).

Number of estimated parameters	7
Sample size	889
Init log likelihood	-5937.274
Final log likelihood	-508.6166
Akaike Information Criterion	1031.233
Bayesian Information Criterion	1064.764
Optimization time	0:00:01.416986

Description	Coeff. estimate	Robust Asympt. std. error	t-stat	p-value
choice_scale_parameter	0.0782	0.0187	4.18	2.97e-05
choice_asc_car	-3.07	5.03	-0.609	0.542
choice_asc_pt	-8.71	5.90	-1.48	0.140
choice_beta_time_pt	-12.2	4.26	-2.86	0.00424
choice_beta_time_car	-26.0	8.84	-2.94	0.00328
choice_beta_dist_work	-2.61	0.840	-3.11	0.00186
choice_beta_dist_other_purposes	-4.12	1.22	-3.38	0.000732

Table 2: Estimation results for the baseline mode choice model

The estimation results reported in Table 2 are consistent with expectations. The estimated coefficients associated with travel time and distance are all negative and statistically significant, indicating that longer travel times and greater distances reduce utility. Because the cost coefficient is fixed to -1, the time coefficients can be interpreted as values of time expressed in monetary units. Their magnitudes suggest a larger value of time for car users than for public transport users. Distance has a significant effect on the utility of slow modes, with a stronger impact for non-work trips than for work trips. Finally, the estimated scale parameter is positive and statistically significant, confirming the validity of the chosen normalization.

4.3 Example 2: MIMIC model in the continuous case

We now introduce the first latent-variable specification of the case study. In contrast with Example 4.2, this model contains no choice component. It is therefore a pure *Multiple Indicators Multiple Causes* (MIMIC) model, as described in Section 3.1. Its purpose is to introduce the latent variable, its structural equation, the associated measurement equations, and the normalization strategy, without the additional complexity of a choice model.

Although this example contains no choice likelihood, it uses the common data-preparation function `read_data`. The resulting sample is therefore also subject to the choice-oriented filters used in the other examples. This ensures that the MIMIC and hybrid models are estimated on the same observations, but it means that the MIMIC example is not an analysis of all respondents with valid psychometric indicators.

In this example, a single latent variable is considered, $x^*_{n,\text{car}}$, interpreted as a *car-centric attitude*. The structural equation follows the linear specification of Eq. (2):

$$\begin{aligned} x^*_{n,\text{car}} = \psi_0 &+ \psi_1 \,\text{HighEducation}_n + \psi_2 \,\text{TopManager}_n \\ &+ \psi_3 \,\text{CarOrientedParents}_n + \psi_4 \,\text{LowEducation}_n \\ &+ \psi_5 \,\text{UsedToGoToSchoolByCar}_n + \sigma_{\omega,\text{car}} \,\omega_n. \end{aligned} \tag{29}$$

where $\omega_n \sim N(0, 1)$. The explanatory variables included in the structural equation are exactly those declared in the semantic specification used by the script: `top_manager`, `car_oriented_parents`, `high_education`, `low_education`, and `used_to_go_to_school_by_car`. As in the implementation, the error scale $\sigma_{\omega,\text{car}}$ is estimated and constrained to be positive.

The latent variable $x^*_{n,\text{car}}$ is measured using the following Likert indicators:

`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.

`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.

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

In the Optima data, the valid Likert responses use the labels $1, \ldots, 5$, while 6 and -1 denote non-informative responses.

All these indicators are treated as ordinal variables defined on a common symmetric Likert scale with five categories. They provide indirect observations of the latent construct $x^*_{n,\text{car}}$, which is not directly observed and must be inferred through the measurement equations introduced in Section 1.3.

Although these indicators are observed on ordinal response scales, they are treated here according to the *continuous case* introduced in Section 1.2. This is a deliberate simplification for pedagogical purposes: it allows us to focus first on the latent variable itself and on its identification, before introducing ordinal measurement equations in later examples.

For each indicator ℓ in the above set, the measurement equation is written as

$$I_{n\ell} = \lambda_{\ell 0} + \lambda_{\ell 1} x^*_{n,\text{car}} + \sigma_{\upsilon\ell} \upsilon_{n\ell}, \tag{30}$$

where $\upsilon_{n\ell} \sim N(0, 1)$. Therefore, conditional on the latent variable, the contribution of the indicators to the likelihood is

$$\Lambda^I_n\left(I_n \mid x^*_{n,\text{car}}, x_n\right) = \prod_{\ell=1}^{L_n} \frac{1}{\sigma_{\upsilon\ell}} \phi\left(\frac{I_{n\ell} - \lambda_{\ell 0} - \lambda_{\ell 1} x^*_{n,\text{car}}}{\sigma_{\upsilon\ell}}\right), \tag{31}$$

where $\phi(\cdot)$ denotes the standard normal density.

Because the latent variable is not observed, the likelihood contribution of individual n is obtained by integrating out $x^*_{n,\text{car}}$:

$$\Lambda^{\text{MIMIC}}_n = \int_{x^*_{n,\text{car}}} \Lambda^I_n\left(I_n \mid x^*_{n,\text{car}}, x_n\right) f\left(x^*_{n,\text{car}} \mid x_n\right) dx^*_{n,\text{car}}, \tag{32}$$

where $f(x^*_{n,\text{car}} \mid x_n; \alpha)$ is the conditional density implied by the structural equation (29). As discussed in Section 1.2, this integral has no closed form and is evaluated by Monte-Carlo integration[1].

[1] For the particular linear-Gaussian MIMIC specification used here, the integral can in fact be evaluated analytically because the marginal distribution of the indicators is multivariate normal. The implementation nevertheless evaluates it by Monte Carlo in order to use the same generic construction as the later hybrid choice models, for which no such closed form is generally available.

Normalization. As explained in Section 2, identification of the latent variable requires fixing its location and its scale. In the implementation, this is achieved by adopting a reference-indicator strategy based on indicator `Envir01`. Its intercept is fixed to zero:

$$\lambda_{\texttt{Envir01},0} = 0, \tag{33}$$

which anchors the location of the latent-variable scale. Its loading on the latent variable is fixed to -1:

$$\lambda_{\texttt{Envir01},1} = -1, \tag{34}$$

which anchors the scale and the orientation of the latent variable. With this sign convention, larger values of the car-centric attitude correspond to lower expected values of `Envir01`. Once these two constraints are imposed, the remaining intercepts and loadings are interpreted relative to the reference indicator.

This specification is implemented in the script `plot_h02_lv_mimic_gauss.py`. The semantic specification of the latent variable itself is defined separately in `one_latent_variable_spec.py`; the example script then adds the continuous measurement equations, the normalization constraints, and the maximum-likelihood estimation setup. The resulting model is the first example in the tutorial where the latent-variable machinery is used explicitly.

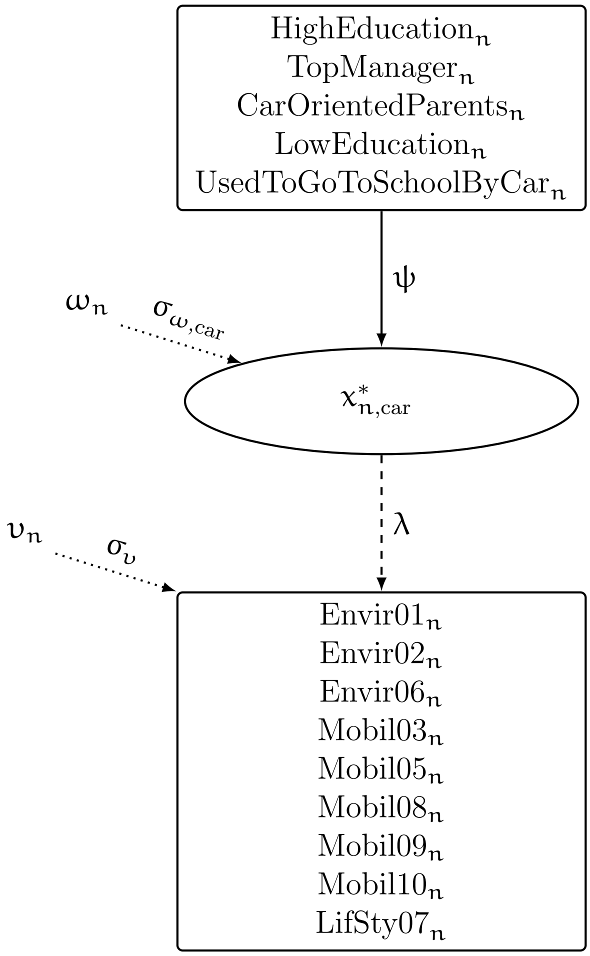

Figure 2: Graphical representation of the MIMIC model for the car-centric latent variable.

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
measurement_Envir01_sigma_log	0.00969	0.0450	0.215	0.829

measurement_intercept_Envir02	1.76	0.160	11.0	0.00
measurement_coefficient_car_centric_attitude_Envir02	-0.597	0.0574	-10.4	0.00
measurement_Envir02_sigma_log	0.0247	0.0271	0.910	0.363
measurement_intercept_Envir06	3.30	0.101	32.8	0.00
measurement_coefficient_car_centric_attitude_Envir06	-0.376	0.0385	-9.77	0.00
measurement_Envir06_sigma_log	-0.303	0.0183	-16.5	0.00
measurement_intercept_Mobil03	2.86	0.156	18.3	0.00
measurement_coefficient_car_centric_attitude_Mobil03	-0.150	0.0550	-2.73	0.00629
measurement_Mobil03_sigma_log	0.289	0.0286	10.1	0.00
measurement_intercept_Mobil05	2.42	0.172	14.0	0.00
measurement_coefficient_car_centric_attitude_Mobil05	-0.339	0.0607	-5.59	2.25e-08
measurement_Mobil05_sigma_log	0.320	0.0288	11.1	0.00
measurement_intercept_Mobil08	3.22	0.150	21.4	0.00
measurement_coefficient_car_centric_attitude_Mobil08	0.336	0.0550	6.11	9.73e-10
measurement_Mobil08_sigma_log	0.141	0.0255	5.52	3.36e-08
measurement_intercept_Mobil09	2.95	0.133	22.1	0.00
measurement_coefficient_car_centric_attitude_Mobil09	-0.351	0.0498	-7.05	1.80e-12
measurement_Mobil09_sigma_log	0.0575	0.0196	2.94	0.00329
measurement_intercept_Mobil10	5.30	0.266	19.9	0.00
measurement_coefficient_car_centric_attitude_Mobil10	0.533	0.0896	5.95	2.73e-09
measurement_Mobil10_sigma_log	0.594	0.0428	13.9	0.00
measurement_intercept_LifSty07	2.49	0.141	17.7	0.00
measurement_coefficient_car_centric_attitude_LifSty07	0.0866	0.0488	1.77	0.0761
measurement_LifSty07_sigma_log	0.217	0.0247	8.76	0.00

Table 5: Estimation results for the MIMIC model (continuous measurement): measurement equations

The estimation results for the structural equation are reported in Table 4. The estimated intercept is negative and highly significant, indicating a low average level of the car-centric attitude in the population under the chosen normalization. Among the explanatory variables, having car-oriented parents is associated with a significantly higher car-centric attitude, whereas a high level of education is associated with a significantly lower one. The remaining variables, namely `top_manager`, `low_education`, and `used_to_go_to_school_by_car`, are not statistically significant in this specification. The estimated scale parameter of the structural disturbance is close to one, as $\exp(-0.00271) \approx 1$, indicating that a substantial share of the variability of the latent attitude remains unexplained by the observed socio-demographic characteristics.

The estimation results for the measurement equations are reported in Table 5. With the exception of `LifSty07`, all indicator loadings are statistically significant and therefore provide information about the latent construct. The signs of the coefficients are consistent with the normalization adopted through the reference indicator `Envir01`: larger values of the car-centric attitude are associated with lower expected values of the environmental and public-transport-related indicators (`Envir02`, `Envir06`, `Mobil05`, and `Mobil09`), while they are associated with higher expected values of indicators reflecting greater discomfort with public transport or a stronger attachment to private mobility (`Mobil08` and `Mobil10`). The loading of `Mobil03` is statistically significant but comparatively small, suggesting a weaker relationship with the

Number of estimated parameters	32
Sample size	889
Init log likelihood	-26577.53
Final log likelihood	-13029.98
Akaike Information Criterion	26123.95
Bayesian Information Criterion	26277.24
Number of draws	50000
Optimization time	0:25:49.527588

Table 3: Estimation results for the MIMIC model (continuous measurement): summary statistics

Description	Coeff. estimate	Asympt. std. error	t-stat	p-value
struct_car_centric_attitude_intercept	-2.64	0.115	-23.0	0.00
struct_car_centric_attitude_top_manager	0.0914	0.143	0.638	0.523
struct_car_centric_attitude_car_oriented_parents	0.237	0.102	2.32	0.0203
struct_car_centric_attitude_high_education	-0.541	0.130	-4.15	3.27e-05
struct_car_centric_attitude_low_education	0.123	0.125	0.983	0.326
struct_car_centric_attitude_used_to_go_to_school_by_car	0.0998	0.560	0.178	0.859
struct_car_centric_attitude_sigma_log	-0.00271	0.0740	-0.0366	0.971

Table 4: Estimation results for the MIMIC model (continuous measurement): structural equation

latent construct. Overall, the estimated loadings confirm that the indicators capture a common underlying attitudinal dimension that can reasonably be interpreted as a car-centric attitude.

4.4 Example 3: hybrid mode choice model with sequential estimation

We now extend the baseline mode choice model by introducing the latent variable estimated in Section 4.3 into the utility specification. This is our first hybrid choice model, estimated using a *sequential* approach.

Let $x^*_{n,\text{car}}$ denote the car-centric latent variable introduced in Section 4.3. As in the selected implementation, its structural equation is not re-estimated jointly with the choice model. Instead, the structural parameters estimated in the MIMIC model are treated as fixed and the latent variable is specified as

$$\begin{aligned} x^*_{n,\text{car}} = {} & \hat{\psi}_0 + \hat{\psi}_1 \text{HighEducation}_n + \hat{\psi}_2 \text{TopManager}_n \\ & + \hat{\psi}_4 \text{CarOrientedParents}_n + \hat{\psi}_5 \text{LowEducation}_n + \hat{\psi}_6 \text{UsedToGoToSchoolByCar}_n \\ & + \hat{\sigma}_{\omega,\text{car}}\, \omega_n, \end{aligned} \tag{35}$$

where $\omega_n \sim N(0,1)$. Therefore, although the structural parameters are fixed, the latent variable remains random through the structural disturbance term.

The systematic utilities are then augmented with this latent variable. The specifications for public transport and slow modes remain unchanged with respect to Example 4.2. For the car alternative, the utility function explicitly incorporates the latent variable as follows:

$$V_{\text{Car},n} = \beta_{\text{asc,Car}} + \beta_{\text{time,Car}}\,\text{TimeCar}_n + \beta_{\text{cost}}\,\text{CostCar}_n + \beta_{\text{car,LV}}\,x^*_{n,\text{car}}, \tag{36}$$

where $\beta_{\text{car,LV}}$ captures the impact of the car-centric attitude on the utility of the car alternative.

The conditional choice probability keeps the standard logit form:

$$P(i_n \mid x_n, x^*_{n,\text{car}}; \beta) = \frac{\exp\big(V(x_{i_n,n}, x^*_{n,\text{car}}; \beta)\big)}{\sum_{j\in\mathcal{C}_n}\exp\big(V(x_{jn}, x^*_{n,\text{car}}; \beta)\big)}. \tag{37}$$

Because the latent variable is random, the individual likelihood contribution involves integration over its distribution:

$$\Lambda_n(\beta) = \int_{x^*_{n,\text{car}}} P(i_n \mid x_n, x^*_{n,\text{car}}; \beta)\, f(x^*_{n,\text{car}} \mid x_n; \alpha)\, dx^*_{n,\text{car}}, \tag{38}$$

and the log-likelihood is

$$\mathcal{L}(\beta) = \sum_n \ln\left[\int_{x^*_{n,\text{car}}} P(i_n \mid x_n, x^*_{n,\text{car}}; \beta)\, f(x^*_{n,\text{car}} \mid x_n; \alpha)\, dx^*_{n,\text{car}}\right]. \tag{39}$$

In this specification, the indicators used in the previous example do not appear explicitly in the likelihood. They intervene only indirectly through the previously estimated structural parameters $\hat{\psi}$. This is the defining feature of the sequential approach: the latent variable is first estimated from the measurement model, and then introduced into the choice model without joint estimation.

A consequence of the sequential procedure is that uncertainty from the first-stage MIMIC estimation is not propagated into the second-stage standard errors. The second-stage covariance matrix is conditional on the estimated structural and measurement parameters from the first stage. Sequential estimates can therefore be useful for illustration or initialization, but they should not be interpreted as equivalent to the estimates and standard errors from a jointly estimated hybrid choice model.

This model is implemented in the script `plot_h03_mode_lv_gauss_seq.py`. The script reads the results of the MIMIC model, reconstructs the latent variable, inserts it into the utility functions, and estimates the choice model by maximum likelihood.

While this approach is simple and useful for illustration, it does not correspond to the full hybrid likelihood of Eq. (21), as the measurement equations are not estimated jointly with the choice model. The next example will address this limitation by moving to simultaneous estimation.

Figure 3 provides a graphical representation of the sequential hybrid choice model with one latent variable. The figure combines the choice model (left) and the latent variable model (right), together with the link between them.

On the left-hand side, the observed explanatory variables x_n, listed in the upper node, enter the specification of the random utility functions $U_{\text{PT},n}$, $U_{\text{Car},n}$, and $U_{\text{Slow},n}$. This corresponds to the structural part of the choice model, with parameters β. The random utilities are also affected by an unobserved disturbance ε_n, scaled by the parameter μ. The observed choice i_n results from the selection of the alternative with the highest utility, represented by the mapping $i_n = \arg\max_{j\in\mathcal{C}_n} U_{jn}$.

On the right-hand side, the latent variable $x^*_{n,\text{car}}$ is specified through a structural equation as a function of observed characteristics, with parameters $\hat{\psi}$. It is also affected by a random term ω_n, scaled by $\hat{\sigma}_{\omega,\text{car}}$. In the sequential estimation approach, these parameters are estimated in a first step using the measurement model and are treated as fixed in the subsequent choice model estimation.

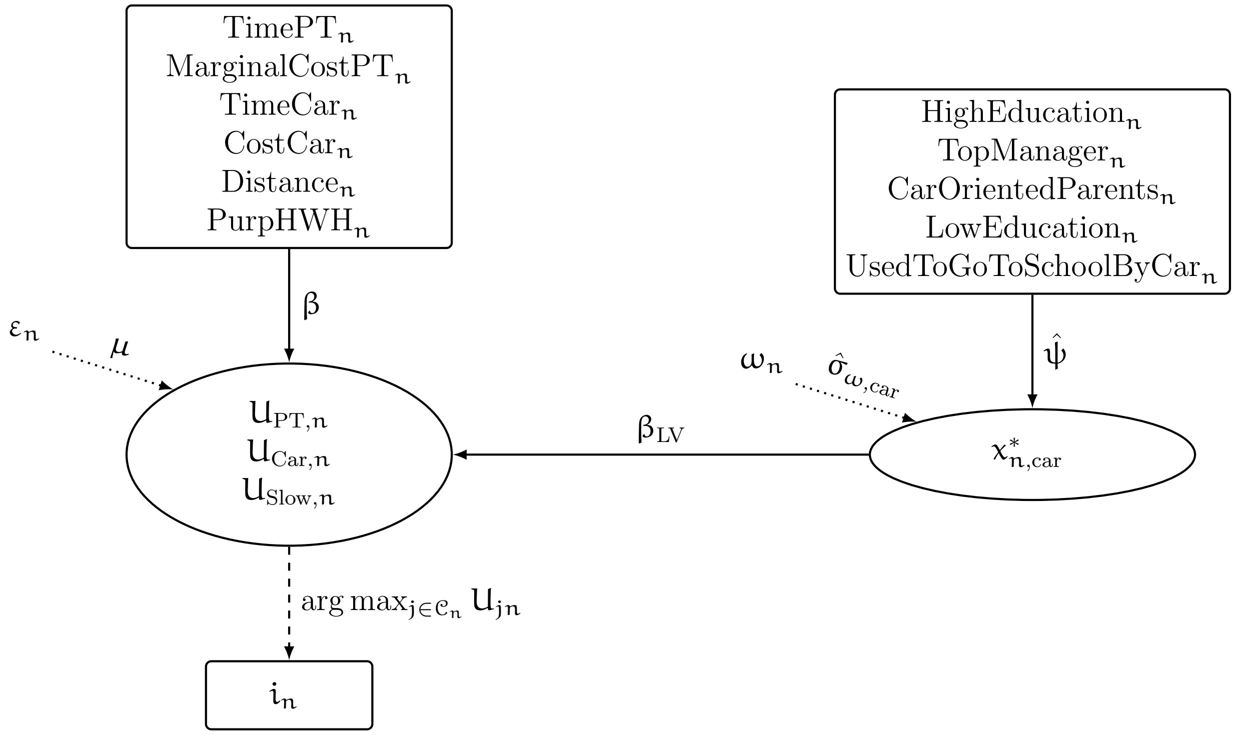

Figure 3: Graphical representation of the sequential hybrid choice model with one latent variable.

The horizontal link between the latent variable and the utility functions captures the role of attitudes in the choice process. The latent variable enters the utility specification through the parameter β_{LV}, thereby influencing the relative attractiveness of the alternatives. This representation highlights the two-stage nature of the sequential approach: the latent variable is first constructed from its structural equation, and then incorporated as an explanatory factor in the choice model.

Number of estimated parameters	8
Sample size	889
Init log likelihood	-5607.637
Final log likelihood	-507.5701
Akaike Information Criterion	1031.14
Bayesian Information Criterion	1069.461
Number of draws	50000
Optimization time	0:00:53.058620

Table 6: Estimation results for the hybrid choice model with sequential estimation: summary statistics

The estimation results are reported in Table 7. The coefficients associated with travel time and distance remain remarkably stable compared to those obtained for the baseline mode choice model in Table 2. They retain the same signs, magnitudes, and levels of statistical significance, indicating that the introduction of the latent variable does not substantially modify the effects of the observed travel attributes. As in the baseline model, the implied value of time is larger for car than for public transport, and distance has a stronger impact on the utility of slow modes for non-work trips than for work trips. The coefficient associated with the car-centric latent variable is positive, implying that individuals with a stronger car-oriented attitude tend to derive greater utility from the car alternative. However, this effect is only marginally significant ($p = 0.101$), providing limited statistical evidence that the latent attitude contributes to explaining mode choice beyond the observed socio-demographic and travel attributes. The estimated scale parameter remains virtually unchanged from the baseline specification.

Description	Coeff. estimate	Robust Asympt. std. error	t-stat	p-value
choice_scale_parameter	0.0782	0.0190	4.12	3.80e-05
choice_asc_pt	-8.79	5.95	-1.48	0.140
choice_beta_time_pt	-12.3	4.34	-2.84	0.00457
choice_asc_car	-2.65	5.04	-0.527	0.598
choice_beta_time_car	-26.2	9.01	-2.91	0.00361
choice_beta_car_centric_attitude_car	3.80	2.32	1.64	0.101
choice_beta_dist_work	-2.62	0.845	-3.10	0.00195
choice_beta_dist_other_purposes	-4.14	1.24	-3.35	0.000813

Table 7: Estimation results for the hybrid choice model with sequential estimation

4.5 Example 4: hybrid mode choice model with simultaneous estimation

We now move from the sequential approach of Example 4.4 to the full hybrid specification described in Section 3.2. The behavioral content of the model is unchanged: we consider the same mode choice model, the same car-centric latent variable, the same structural equation, and the same continuous measurement equations as in Section 4.3. What changes is the estimation strategy. Instead of estimating the latent-variable model first and then introducing the latent variable into the choice model, all components are now estimated jointly.

The latent variable $x^*_{n,\text{car}}$ is specified exactly as in Section 4.3, with the same structural equation and the same normalization. Likewise, the measurement equations of the indicators are the continuous measurement equations introduced in Section 1.2 and used in Example 4.3. The utility specification is the same as in Example 4.4, in particular for the car alternative:

$$V_{\text{Car},n} = \beta_{\text{asc,Car}} + \beta_{\text{time,Car}}\,\text{TimeCar}_n + \beta_{\text{cost}}\,\text{CostCar}_n + \beta_{\text{car,LV}}\,x^*_{n,\text{car}}. \tag{40}$$

The key difference is that the latent-variable parameters and the choice-model parameters are now estimated in one single optimization problem.

For each individual n, the full observation is the pair (i_n, I_n), where i_n is the observed mode choice and I_n is the vector of psychometric indicators. Conditional on the latent variable, the individual contribution to the likelihood is therefore the product of the conditional choice probability and the conditional indicator likelihood:

$$\Lambda_n(i_n, I_n \mid x^*_n, x_n; \beta, \lambda, \tau) = P(i_n \mid x^*_n, x_n; \beta)\Lambda^I_n(I_n \mid x^*_n, x_n; \lambda, \tau), \tag{41}$$

where the first term is the logit choice probability and the second term is the continuous measurement contribution already introduced in Section 4.3. Because the latent variable is not observed, the individual likelihood contribution becomes

$$\Lambda_n = \int_{x^*_{n,\text{car}}} P(i_n \mid x_n, x^*_{n,\text{car}}; \beta)\; \Lambda^I_n(I_n \mid x^*_{n,\text{car}}, x_n)\; f(x^*_{n,\text{car}} \mid x_n; \alpha)\; dx^*_{n,\text{car}}, \tag{42}$$

which is exactly the hybrid choice model likelihood specialized to one latent variable and continuous measurement equations. As before, the integral is evaluated by Monte-Carlo integration.

Figure 4 provides a graphical representation of this model. The left-hand side corresponds to the choice component: observed explanatory variables enter the latent utilities, which in turn determine the observed choice. The right-hand side corresponds to the latent-variable model: observed socio-demographic variables determine the latent variable $x^*_{n,\text{car}}$, which is measured

Number of estimated parameters	40
Sample size	889
Init log likelihood	-32514.8
Final log likelihood	-13512.63
Akaike Information Criterion	27105.25
Bayesian Information Criterion	27296.86
Number of draws	50000
Optimization time	1:29:34.924149

Table 8: Estimation results for the simultaneous hybrid choice model (one latent variable, continuous measurement): summary statistics

through the vector of observed indicators. The horizontal arrow linking $x^*_{n,\text{car}}$ to the utility system emphasizes the defining feature of the hybrid model: the same latent construct is both measured through the psychometric indicators and used as an explanatory factor in the choice model. Unlike the sequential approach, the figure represents here a truly joint model, in the sense that the two blocks contribute simultaneously to the same likelihood function.

This specification is implemented in the script `plot_h04_mode_lv_gauss_simult.py`. The script resolves the semantic specification of the latent variable, builds the continuous measurement model, constructs the mode-choice utilities including the latent-variable term, multiplies the conditional measurement and choice likelihood contributions, and integrates the result over the latent variable. This is the first example in the tutorial where the full hybrid likelihood is estimated directly.

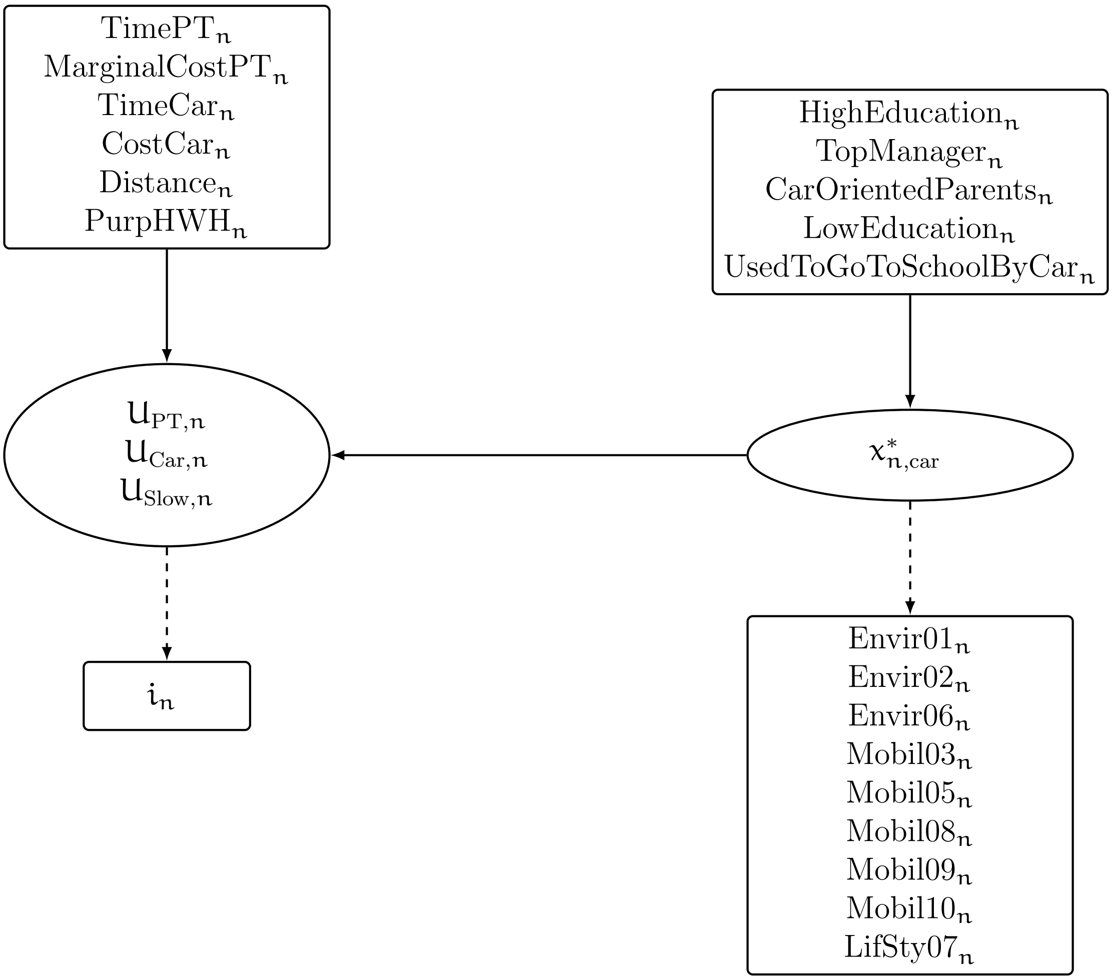

Figure 4: Graphical representation of the hybrid choice model with one latent variable.

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
struct_car_centric_attitude_intercept	-2.66	0.116	-22.9	0.00
struct_car_centric_attitude_top_manager	0.123	0.145	0.852	0.394
struct_car_centric_attitude_car_oriented_parents	0.268	0.104	2.57	0.0102
struct_car_centric_attitude_high_education	-0.530	0.132	-4.01	6.11e-05
struct_car_centric_attitude_low_education	0.122	0.127	0.961	0.336
struct_car_centric_attitude_used_to_go_to_school_by_car	0.119	0.550	0.217	0.828
struct_car_centric_attitude_sigma_log	0.0217	0.0725	0.300	0.764

Table 9: Estimation results for the simultaneous hybrid choice model (one latent variable, continuous measurement): structural equation

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
measurement_Envir01_sigma_log	-0.0157	0.0469	-0.335	0.737
measurement_intercept_Envir02	1.81	0.154	11.7	0.00
measurement_coefficient_car_centric_attitude_Envir02	-0.581	0.0554	-10.5	0.00
measurement_Envir02_sigma_log	0.0262	0.0276	0.950	0.342
measurement_intercept_Envir06	3.35	0.0956	35.1	0.00
measurement_coefficient_car_centric_attitude_Envir06	-0.358	0.0369	-9.70	0.00
measurement_Envir06_sigma_log	-0.295	0.0174	-17.0	0.00
measurement_intercept_Mobil03	2.83	0.153	18.5	0.00
measurement_coefficient_car_centric_attitude_Mobil03	-0.163	0.0538	-3.04	0.00240
measurement_Mobil03_sigma_log	0.287	0.0289	9.94	0.00
measurement_intercept_Mobil05	2.40	0.167	14.4	0.00
measurement_coefficient_car_centric_attitude_Mobil05	-0.345	0.0585	-5.89	3.85e-09
measurement_Mobil05_sigma_log	0.317	0.0290	11.0	0.00
measurement_intercept_Mobil08	3.16	0.144	21.9	0.00
measurement_coefficient_car_centric_attitude_Mobil08	0.314	0.0531	5.92	3.19e-09
measurement_Mobil08_sigma_log	0.145	0.0257	5.65	1.58e-08
measurement_intercept_Mobil09	2.99	0.128	23.3	0.00
measurement_coefficient_car_centric_attitude_Mobil09	-0.336	0.0480	-7.00	2.60e-12
measurement_Mobil09_sigma_log	0.0601	0.0195	3.08	0.00204
measurement_intercept_Mobil10	5.28	0.260	20.3	0.00
measurement_coefficient_car_centric_attitude_Mobil10	0.524	0.0874	6.00	1.99e-09
measurement_Mobil10_sigma_log	0.593	0.0429	13.8	0.00
measurement_intercept_LifSty07	2.42	0.139	17.5	0.00

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
choice_scale_parameter	0.0730	0.00907	8.04	8.88e-16
choice_asc_pt	-10.8	4.17	-2.60	0.00938
choice_beta_time_pt	-12.6	2.78	-4.55	5.39e-06
choice_asc_car	24.1	6.52	3.70	0.000211
choice_beta_time_car	-27.3	4.71	-5.80	6.78e-09
choice_beta_car_centric_attitude_car	9.79	1.95	5.01	5.34e-07
choice_beta_dist_work	-2.78	0.409	-6.81	9.62e-12
choice_beta_dist_other_purposes	-4.44	0.720	-6.16	7.18e-10

Table 11: Estimation results for the simultaneous hybrid choice model (one latent variable, continuous measurement): choice model

measurement_coefficient_car_centric_attitude_LifSty07	0.0626	0.0473	1.32	0.186
measurement_LifSty07_sigma_log	0.218	0.0251	8.69	0.00

Table 10: Estimation results for the simultaneous hybrid choice model (one latent variable, continuous measurement): measurement equations

The estimation results reported in Tables 9, 10, and 11 are broadly consistent with those obtained in the previous examples. The estimates of the structural equation and of the measurement equations remain remarkably stable relative to the MIMIC model of Section 4.3. In particular, the effects of car-oriented parents and high education on the car-centric attitude retain similar magnitudes and levels of significance, and the measurement loadings exhibit the same signs and interpretation as in Table 5. The most notable changes appear in the choice model. The coefficients associated with travel time and distance remain close to those obtained in the baseline and sequential specifications, confirming the robustness of the effects of the observed travel attributes. However, the coefficient associated with the car-centric latent variable increases substantially, from 3.80 in the sequential model (Table 7) to 9.79 in the simultaneous model (Table 11), and becomes highly significant. This result indicates that, once the measurement and choice components are estimated jointly, the latent attitude plays a much more important role in explaining mode choice. The simultaneous estimation therefore reveals a stronger relationship between attitudes and behavior than the sequential approach, illustrating one of the main motivations for estimating the full hybrid choice model.

4.6 Example 5: hybrid mode choice model with ordered-logit measurement

In this example, the hybrid choice model of Example 4.5 is kept unchanged on the choice side and on the latent-variable side, but the measurement model is modified in order to reflect the ordinal nature of the Likert indicators. The structural equation of the latent variable remains the one introduced previously, and the mode choice model is still specified through the hybrid likelihood of Eqs. (19)–(21). The only change concerns the measurement equations entering the term $\Lambda^I_n(I_n \mid x^*_n, x_n)$.

More precisely, the continuous measurement equations used in Example 4.5 are replaced by ordinal measurement equations of the type introduced in Section 1.3. For each indicator ℓ, we introduce an underlying latent response and a system of thresholds as in Eqs. (8)–(10). In the present example, the latent response is modeled as a linear function of the car-centric latent

Number of estimated parameters	41
Sample size	889
Init log likelihood	-20187.62
Final log likelihood	-10505.44
Akaike Information Criterion	21092.88
Bayesian Information Criterion	21289.27
Number of draws	50000
Optimization time	3:02:20.675567

Table 12: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered logit): summary statistics

variable, exactly as in Eq. (11). The contribution of each indicator to the conditional likelihood therefore has the same structure as Eq. (12), except that the normal cumulative distribution is replaced by the logistic cumulative distribution. The overall conditional indicator likelihood is then obtained, as in Eq. (16), by multiplying the contributions of all indicators.

The hybrid model itself remains identical in structure to the one introduced in Example 4.5. In particular, the latent variable still enters the utility of the car alternative, while the other utilities are unchanged. The full likelihood is therefore still given by the hybrid formulation of Eq. (20), with the sole difference that the measurement component now corresponds to an ordered-logit specification rather than to the continuous case.

The main conceptual novelty of this example is therefore not the hybrid structure, but the treatment of the indicators. In the previous example, the Likert responses were approximated as continuous variables. Here, they are modeled explicitly as ordinal outcomes, which is more natural for attitudinal statements recorded on ordered response scales. This requires estimating a threshold structure in addition to the intercepts, loadings, and measurement scales.

The normalization strategy also differs from that of the continuous case. As explained in Section 2, identification requires normalizations both at the latent-variable level and at the ordinal measurement level. The latent variable is normalized exactly as before using the reference-indicator strategy: for indicator `Envir01`, the intercept is fixed to zero and the loading on the latent variable is fixed to -1. In addition, the scale of the ordinal measurement layer is anchored by fixing the measurement scale parameter of the same reference indicator to one. Consequently, all parameters associated with the measurement equation of `Envir01` are fixed and none of them is estimated. The reference indicator therefore serves exclusively to define the location, orientation, and scale of the latent-variable measurement system.

With these constraints, the remaining intercepts, loadings, threshold parameters, and measurement scales are interpreted relative to the chosen reference indicator and to the common ordinal measurement scale.

This specification is implemented in the script `plot_h05_mode_lv_ordlogit_simult.py`. Relative to the previous example, the model structure is unchanged; only the measurement equations and the associated normalization are modified in order to move from the continuous case to an explicit ordinal treatment of the indicators.

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
measurement_intercept_Envir02	1.07	0.154	6.94	4.01e-12
measurement_coefficient_car_centric_attitude_Envir02	-0.525	0.0379	-13.8	0.00
measurement_Envir02_sigma_log	-0.164	0.110	-1.50	0.134
measurement_intercept_Envir06	2.44	0.244	10.0	0.00
measurement_coefficient_car_centric_attitude_Envir06	-0.362	0.0339	-10.7	0.00

measurement_Envir06_sigma_log	-0.431	0.103	-4.19	2.82e-05
measurement_intercept_Mobil03	0.454	0.103	4.42	1.00e-05
measurement_coefficient_car_centric_attitude_Mobil03	-0.175	0.0352	-4.97	6.64e-07
measurement_Mobil03_sigma_log	0.0237	0.105	0.225	0.822
measurement_intercept_Mobil05	0.760	0.144	5.29	1.25e-07
measurement_coefficient_car_centric_attitude_Mobil05	-0.380	0.0434	-8.74	0.00
measurement_Mobil05_sigma_log	0.150	0.111	1.35	0.179
measurement_intercept_Mobil08	-1.13	0.142	-7.95	2.00e-15
measurement_coefficient_car_centric_attitude_Mobil08	0.315	0.0423	7.46	8.90e-14
measurement_Mobil08_sigma_log	0.129	0.109	1.18	0.237
measurement_intercept_Mobil09	1.69	0.192	8.80	0.00
measurement_coefficient_car_centric_attitude_Mobil09	-0.355	0.0338	-10.5	0.00
measurement_Mobil09_sigma_log	-0.207	0.106	-1.96	0.0501
measurement_intercept_Mobil10	0.0872	0.155	0.563	0.573
measurement_coefficient_car_centric_attitude_Mobil10	0.667	0.0779	8.57	0.00
measurement_Mobil10_sigma_log	0.440	0.132	3.32	0.000885
measurement_intercept_LifSty07	-1.29	0.140	-9.21	0.00
measurement_coefficient_car_centric_attitude_LifSty07	0.139	0.0366	3.80	0.000146
measurement_LifSty07_sigma_log	0.147	0.113	1.30	0.194

Table 14: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered logit): measurement equations

The estimation results reported in Tables 13, 14, 15, and 16 confirm the qualitative findings of the previous hybrid specifications. In the structural equation, the same variables remain relevant: individuals with car-oriented parents have a significantly higher car-centric attitude, whereas individuals with high education have a significantly lower one. The estimated measurement coefficients keep the same signs as in the continuous measurement model, but the ordinal specification strengthens the role of the indicators: all loadings are now statistically significant, including `LifSty07`, which was weak in the continuous case. The threshold parameters are also significant, indicating that the ordered-logit measurement layer is well identified. On the choice side, the coefficients of travel time and distance remain close to those obtained in the baseline, sequential, and simultaneous continuous models, confirming the stability of the observed-attribute effects. The coefficient of the car-centric latent variable remains positive and highly significant, although its magnitude is smaller than in the continuous measurement specification. Thus, explicitly treating the Likert indicators as ordinal preserves the main behavioral conclusions while providing a more appropriate measurement model for the psychometric indicators.

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
struct_car_centric_attitude_intercept	0.352	0.243	1.45	0.148
struct_car_centric_attitude_top_manager	0.320	0.294	1.09	0.277
struct_car_centric_attitude_car_oriented_parents	0.706	0.213	3.31	0.000931
struct_car_centric_attitude_high_education	-1.05	0.269	-3.90	9.50e-05
struct_car_centric_attitude_low_education	0.379	0.260	1.46	0.145
struct_car_centric_attitude_used_to_go_to_school_by_car	0.182	1.19	0.153	0.878
struct_car_centric_attitude_sigma_log	0.756	0.0914	8.27	2.22e-16

Table 13: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered logit): structural equation

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
likert_delta_0_log	-0.686	0.0965	-7.11	1.16e-12
likert_delta_1_log	0.513	0.0882	5.82	5.88e-09

Table 15: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered logit): measurement thresholds

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
choice_scale_parameter	0.0699	0.00908	7.70	1.40e-14
choice_asc_pt	-11.5	4.32	-2.66	0.00771
choice_beta_time_pt	-13.3	2.98	-4.47	7.83e-06
choice_asc_car	-4.34	3.61	-1.20	0.230
choice_beta_time_car	-28.6	5.08	-5.62	1.88e-08
choice_beta_car_centric_attitude_car	5.30	1.11	4.77	1.88e-06
choice_beta_dist_work	-2.91	0.436	-6.67	2.51e-11
choice_beta_dist_other_purposes	-4.63	0.762	-6.08	1.21e-09

Table 16: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered logit): choice model

4.7 Example 6: hybrid mode choice model with ordered-probit measurement

This specification is identical to the ordered-logit hybrid model presented in Example 4.6. The structural equation, the utility specification, the hybrid likelihood (Eq. (20)), and the normalization strategy remain unchanged.

The only difference lies in the measurement equations for the Likert indicators. Instead of using the logistic cumulative distribution, the ordinal probabilities are computed using the normal cumulative distribution, as defined in Eq. (12). This leads to an ordered-probit measurement model.

The implementation is provided in the script `plot_h06_mode_lv_ordprobit_simult.py`.

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
measurement_intercept_Envir02	0.838	0.207	4.04	5.26e-05
measurement_coefficient_car_centric_attitude_Envir02	-0.475	0.0585	-8.11	4.44e-16
measurement_Envir02_sigma_log	-0.111	0.184	-0.600	0.548
measurement_intercept_Envir06	1.39	0.252	5.53	3.25e-08
measurement_coefficient_car_centric_attitude_Envir06	-0.319	0.0385	-8.29	2.22e-16
measurement_Envir06_sigma_log	-0.478	0.165	-2.90	0.00371
measurement_intercept_Mobil03	1.02	0.250	4.07	4.72e-05
measurement_coefficient_car_centric_attitude_Mobil03	-0.198	0.0508	-3.91	9.36e-05
measurement_Mobil03_sigma_log	0.320	0.196	1.64	0.101
measurement_intercept_Mobil05	1.43	0.411	3.49	0.000490
measurement_coefficient_car_centric_attitude_Mobil05	-0.512	0.0997	-5.13	2.90e-07
measurement_Mobil05_sigma_log	0.570	0.226	2.51	0.0119
measurement_intercept_Mobil08	0.247	0.122	2.03	0.0421
measurement_coefficient_car_centric_attitude_Mobil08	0.424	0.0816	5.20	2.04e-07
measurement_Mobil08_sigma_log	0.511	0.215	2.38	0.0174
measurement_intercept_Mobil09	1.26	0.260	4.84	1.33e-06
measurement_coefficient_car_centric_attitude_Mobil09	-0.327	0.0447	-7.33	2.29e-13
measurement_Mobil09_sigma_log	-0.153	0.179	-0.853	0.394
measurement_intercept_Mobil10	4.34	2.50	1.74	0.0822
measurement_coefficient_car_centric_attitude_Mobil10	1.90	0.948	2.00	0.0453
measurement_Mobil10_sigma_log	1.57	0.547	2.86	0.00419

Number of estimated parameters	41
Sample size	889
Init log likelihood	-21722.37
Final log likelihood	-10532.96
Akaike Information Criterion	21147.93
Bayesian Information Criterion	21344.32
Number of draws	50000
Optimization time	3:46:12.509410

Table 17: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered probit): summary statistics

measurement_intercept_LifSty07	-0.0495	0.0980	-0.505	0.614
measurement_coefficient_car_centric_attitude_LifSty07	0.156	0.0505	3.08	0.00204
measurement_LifSty07_sigma_log	0.466	0.222	2.10	0.0355

Table 19: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered probit): measurement equations

The estimation results reported in Tables 18, 19, 20, and 21 are qualitatively very similar to those obtained for the ordered-logit specification of Section 4.6. The same explanatory variables remain significant in the structural equation, namely `car_oriented_parents` with a positive effect and `high_education` with a negative effect on the car-centric attitude. Likewise, the measurement equations display the same pattern of signs and significance as in the ordered-logit model, confirming the interpretation of the latent variable and the consistency of the measurement structure. On the choice side, the coefficients associated with travel time, distance, and the scale parameter are almost identical to those obtained previously, demonstrating that the choice component is largely insensitive to the specific distributional assumption adopted for the ordinal indicators. The coefficient of the car-centric latent variable remains positive and highly significant, with an estimated value lying between those obtained for the ordered logit and the continuous measurement specifications. Overall, the ordered-probit and ordered-logit models lead to essentially the same behavioral conclusions. This similarity is expected, as the logistic and normal cumulative distributions have comparable shapes, and it illustrates the well-known empirical robustness of ordinal models with respect to the choice between logit and probit formulations.

The directory also contains `plot_h07_mode_lv_ordprobit_simult_generated.py`. This is not a new behavioral specification. It is an explicit, automatically-generated Biogeme implementation of the ordered-probit model in Example 6. It is useful for inspecting the complete expression tree and for verifying that the higher-level semantic specification produces the intended low-level model. The two scripts should therefore be understood as alternative implementations of the same model, rather than as two additional empirical examples.

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
struct_car_centric_attitude_intercept	-0.270	0.156	-1.73	0.0829
struct_car_centric_attitude_top_manager	0.154	0.183	0.841	0.401
struct_car_centric_attitude_car_oriented_parents	0.441	0.132	3.33	0.000858
struct_car_centric_attitude_high_education	-0.646	0.165	-3.91	9.26e-05
struct_car_centric_attitude_low_education	0.245	0.161	1.52	0.128
struct_car_centric_attitude_used_to_go_to_school_by_car	0.130	0.705	0.185	0.853
struct_car_centric_attitude_sigma_log	0.293	0.0850	3.45	0.000563

Table 18: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered probit): structural equation

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
likert_delta_0_log	-2.48	0.273	-9.06	0.00
likert_delta_1_log	-0.278	0.152	-1.83	0.0668

Table 20: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered probit): measurement thresholds

Description	Coeff. estimate	BHHH Asympt. std. error	t-stat	p-value
choice_scale_parameter	0.0699	0.00907	7.71	1.29e-14
choice_asc_pt	-11.4	4.32	-2.64	0.00822
choice_beta_time_pt	-13.3	2.97	-4.47	7.83e-06
choice_asc_car	-0.245	3.67	-0.0669	0.947
choice_beta_time_car	-28.5	5.06	-5.62	1.87e-08
choice_beta_car_centric_attitude_car	8.20	1.69	4.85	1.24e-06
choice_beta_dist_work	-2.90	0.436	-6.67	2.62e-11
choice_beta_dist_other_purposes	-4.62	0.760	-6.07	1.24e-09

Table 21: Estimation results for the simultaneous hybrid choice model (one latent variable, ordered probit): choice model

5 Conclusion

This document has introduced the specification and estimation of hybrid choice models using Biogeme, combining structural equations for latent variables, measurement equations for indicators, and discrete choice models within a unified framework.

Hybrid choice models extend traditional discrete choice analysis by incorporating latent constructs that capture attitudes and perceptions. By linking these constructs simultaneously to observed indicators and to choices, they provide a richer and more behaviorally meaningful representation of decision-making processes.

A key aspect of these models is identification. Because latent variables and ordinal measurement systems are not directly observed, appropriate normalization is essential. The reference-indicator strategy, together with consistent normalization of ordinal thresholds, offers a practical and interpretable solution.

The case study has illustrated how these elements can be implemented step by step. In particular, the comparison between sequential and simultaneous estimation highlights the importance of accounting for the uncertainty of latent variables. The comparison between sequential and simultaneous estimation highlights an important distinction in the treatment of latent variables. Sequential estimation treats the first-stage parameters as fixed in the choice model, whereas simultaneous estimation estimates the measurement and choice components through one joint likelihood. The two approaches can therefore produce different estimates. Simultaneous estimation provides a coherent joint framework, but its results remain dependent on the identification, distributional, and measurement assumptions of the model and are not automatically more behaviorally valid.

Finally, the transition from continuous to ordinal measurement models emphasizes the importance of aligning the specification with the nature of the data. While ordinal models are more demanding computationally, they provide a more appropriate representation of Likert-scale indicators.

For completeness, Appendix A provides a description of all variables used in the case study, including the explanatory variables entering the structural and choice models as well as the psychometric indicators. Appendix B contains the complete specification files corresponding to the examples presented throughout the document. These files illustrate how the theoretical concepts discussed in the main text are translated into executable Biogeme specifications and can serve as templates for the development of new hybrid choice models.

Overall, Biogeme offers a flexible and powerful environment for estimating hybrid choice models, enabling researchers to progressively build and analyze complex behavioral specifications.

A 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 22: Description of variables

B 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 3.3.3`. It is possible that minor adaptations of the syntax may be required for future versions of Biogeme.

B.1 choice_latent_variables .py

```python
"""Choice model specification used in the latent-variable tutorial.

This module defines the systematic utility functions of the mode-choice
model. It illustrates how a choice model is specified independently of the
latent-variable model and how latent variables can subsequently be injected
into the utility functions.

The specification considers three transportation alternatives:

- public transport,
- car,
- slow modes.

The utilities are generated by the function
``generate_utility_functions`` and returned as a dictionary indexed by
alternative identifiers. The resulting dictionary can be passed directly
to a Biogeme logit model.

The example is intentionally structured in two layers:

1. A baseline specification based exclusively on observed attributes
   (travel time, travel cost, distance, and trip purpose).
2. An extended specification in which latent variables may enter the
   utility functions through interaction terms.

The latent variables themselves are not defined in this module. They are
constructed elsewhere from structural and measurement equations and are
provided here as Biogeme expressions through the argument
``latent_expressions``. This separation reflects the conceptual structure
of hybrid choice models, where latent variables are first generated and
then incorporated into the choice model.

In the current example, the latent variable
``car_centric_attitude`` may influence the utility of the car
alternative. If no latent expressions are provided, the function reduces
to the baseline mode-choice specification.

The cost coefficient is normalized to -1 and a positive scale parameter is
estimated. This normalization defines the unit of utility while allowing
the overall scale of the choice model to be estimated.

Michel Bierlaire
Fri Jun 12 2026, 15:00:22
"""

from optima import (
    CostCarCHF,
    MarginalCostPT,
    PurpHWH,
    TimeCar_hour,
    TimePT_hour,
    car_is_available,
    distance_km,
)

from biogeme.expressions import (
    Beta,
    Expression,
)


def generate_utility_functions(
    latent_expressions: dict[str, Expression] | None = None,
) -> dict[int, Expression]:
    """Generate the systematic utility functions of the choice model.

    The function creates the utility expressions associated with the three
    transportation alternatives considered in the tutorial:

    - alternative 0: public transport,
    - alternative 1: car,
    - alternative 2: slow modes.

    The utilities are expressed using Biogeme expressions and include the
    estimated parameters declared in this function.

    Baseline specification
    ----------------------

    Public transport:

    .. math::

       V_{\mathrm{PT}}
       = \beta_{\mathrm{asc,PT}}
       + \beta_{\mathrm{time,PT}}\,\mathrm{TimePT}
       + \beta_{\mathrm{cost}}\,\mathrm{MarginalCostPT}

    Car:

```

```
    .. math::

       V_{\mathrm{Car}}
       = \beta_{\mathrm{asc,Car}}
       + \beta_{\mathrm{time,Car}}\,\mathrm{TimeCar}
       + \beta_{\mathrm{cost}}\,\mathrm{CostCar}

    Slow modes:

    .. math::

       V_{\mathrm{Slow}}
       = \beta_{\mathrm{dist}}\,\mathrm{Distance}

    where the distance coefficient depends on trip purpose.

    Latent-variable extension
    -------------------------

    When ``latent_expressions`` is provided, the latent variable
    ``car_centric_attitude`` enters the utility of the car alternative
    through an additional coefficient:

    .. math::

       V_{\mathrm{Car}}
       \leftarrow
       V_{\mathrm{Car}}
       + \beta_{\mathrm{car\_centric}}
         x^*_{\mathrm{car\_centric}}

    This illustrates the standard mechanism used in hybrid choice models:
    latent variables are treated as additional explanatory variables in the
    utility specification.

    Utility scaling
    ---------------

    After the utilities are constructed, they are multiplied by a positive
    scale parameter. Because the cost coefficient is fixed to -1, the scale
    parameter can be estimated and captures the overall sensitivity of
    choice probabilities to utility differences.

    :param latent_expressions: Optional dictionary mapping latent-variable
        names to Biogeme expressions. If ``None``, the baseline choice model
        is generated. The current specification expects the key
        ``'car_centric_attitude'`` when latent variables are used.
    :return: Dictionary mapping alternative identifiers to systematic
        utility expressions.
    """
    work_trip = PurpHWH == 1
    other_trip_purposes = PurpHWH != 1

    # Choice model: parameters
    choice_beta_cost = Beta('choice_beta_cost', -1, None, 0, 1)

    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
    )

    scale_parameter = Beta('choice_scale_parameter', 1, 0.0001, None, 0)

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

    choice_beta_car_centric_attitude_car = Beta(
        'choice_beta_car_centric_attitude_car', 0, None, None, 0
    )
    v_public_transport_det = (
        choice_asc_pt
        + choice_beta_time_pt * TimePT_hour
        + choice_beta_cost * MarginalCostPT
    )
    v_public_transport = v_public_transport_det

    v_car_det = (
        choice_asc_car
        + choice_beta_time_car * TimeCar_hour
        + choice_beta_cost * CostCarCHF
    )
    v_car = (
        v_car_det
        if latent_expressions is None
        else v_car_det
        + choice_beta_car_centric_attitude_car
        * latent_expressions['car_centric_attitude']
    )

    v_slow_modes_det = choice_beta_dist * distance_km
    v_slow_modes = v_slow_modes_det

    v = {
        0: scale_parameter * v_public_transport,
        1: scale_parameter * v_car,
        2: scale_parameter * v_slow_modes,
    }
    return v
```

```python


def generate_availability():
    return {
        0: 1,  # public transport
        1: car_is_available,  # car
        2: 1,  # slow modes
    }
```

The file choice_latent_variables .py defines the utility specification used by the mode-choice component of the examples. Its main role is to keep the choice model separated from the latent-variable model. The function generate_utility_functions returns the systematic utilities of the three alternatives: public transport, car, and slow modes.

The baseline specification uses only observed travel attributes: travel time and marginal cost for public transport, travel time and cost for car, and distance for slow modes. The distance coefficient is allowed to depend on the trip purpose, distinguishing work trips from other trips.

The module also prepares the hybrid extension of the choice model. If a dictionary of latent-variable expressions is provided, the car-centric latent variable is added to the utility of the car alternative. If no such dictionary is provided, the same function generates the baseline logit model without latent variables. This design makes it possible to use the same utility specification in the standard choice model, in the sequential hybrid model, and in the simultaneous hybrid models.

The cost coefficient is fixed to -1, which defines the utility unit. A positive scale parameter is then estimated and multiplies all systematic utilities. This reflects the normalization strategy discussed earlier: the cost normalization fixes the scale of utility, while the estimated scale parameter controls the sensitivity of the choice probabilities to utility differences.

The value specified in number_of_draws.py overrides the general Monte Carlo default in `biogeme.toml` for these examples. Maximum-likelihood latent-variable specifications use the `NORMAL_MLHS_ANTI` draw type by default: modified Latin hypercube sampling with antithetic draws. The draw type is part of the numerical specification and should be reported together with the number of draws.

B.2 likert_spec .py

```python
"""
Likert indicators and Likert types (SPEC ONLY)

Pure semantic indicator declarations
Michel Bierlaire
Sat Mar 14 2026, 16:26:11
"""

from biogeme.latent_variables import LikertIndicator, LikertType

likert_indicators = [
    LikertIndicator(
        name="Envir01",
        statement="Fuel price should be increased to reduce congestion and air pollution.",
        type_name="likert",
    ),
    LikertIndicator(
        name="Envir02",
        statement="More public transportation is needed, even if taxes are set to pay the additional costs.",
        type_name="likert",
    ),
    LikertIndicator(
        name="Envir03",
        statement="Ecology disadvantages minorities and small businesses.",
        type_name="likert",
    ),
    LikertIndicator(
        name="Envir04",
        statement="People and employment are more important than the environment.",
        type_name="likert",
    ),
    LikertIndicator(
        name="Envir05",
        statement="I am concerned about global warming.",
        type_name="likert",
    ),
    LikertIndicator(
        name="Envir06",
        statement="Actions and decision making are needed to limit greenhouse gas emissions.",
        type_name="likert",
    ),
    LikertIndicator(
        name="Mobil03",
        statement="I use the time of my trip in a productive way.",
        type_name="likert",
    ),
    LikertIndicator(
        name="Mobil05",
        statement="I reconsider frequently my mode choice.",
```

```
            type_name="likert",
        ),
        LikertIndicator(
            name="Mobil08",
            statement="I do not feel comfortable when I travel close to people I do not know.",
            type_name="likert",
        ),
        LikertIndicator(
            name="Mobil09",
            statement="Taking the bus helps making the city more comfortable and welcoming.",
            type_name="likert",
        ),
        LikertIndicator(
            name="Mobil10",
            statement="It is difficult to take the public transport when I travel with my children.",
            type_name="likert",
        ),
        LikertIndicator(
            name="Mobil12",
            statement="It is very important to have a beautiful car.",
            type_name="likert",
        ),
        LikertIndicator(
            name="LifSty01",
            statement="I always choose the best products regardless of price.",
            type_name="likert",
        ),
        LikertIndicator(
            name="LifSty07",
            statement="The pleasure of having something beautiful consists in showing it.",
            type_name="likert",
        ),
]

likert_types = [
    LikertType(
        type_name="likert",
        symmetric=True,
        categories=[1, 2, 3, 4, 5],
        neutral_labels=[6, -1],
    ),
]
```

The file likert_spec.py contains the semantic declaration of the Likert indicators used in the measurement equations. Each indicator is represented by a LikertIndicator object, which associates an internal variable name, such as Envir01 or Mobil10, with the corresponding survey statement. These declarations do not yet specify a statistical measurement equation. They only define the set of observed psychometric indicators that can later be linked to the latent variable.

The file also defines the Likert response scale through a LikertType object. In this example, all indicators share the same five-point symmetric scale, with response categories 1 to 5. The labels 6 and −1 are declared as neutral or non-informative responses, so that they can be treated separately when constructing the measurement likelihood. This separation between indicator declarations and the statistical measurement model reflects the semantic organization used throughout the examples: the present file defines what is observed, while the subsequent model scripts specify how these observations enter the likelihood.

B.3 number_of_draws.py

```
"""Number of Monte-Carlo draws used by the hybrid choice model examples.

This module contains a single constant shared across several examples.
It is intentionally isolated so that the number of draws can be modified
in one place and propagated consistently throughout the examples.

Michel Bierlaire
Wed Jun 17 2026, 13:28:01
"""

NUMBER_OF_DRAWS = 50_000
```

The file number_of_draws.py centralizes the number of draws used in the Monte-Carlo integration. It defines a single constant, NUMBER_OF_DRAWS, set to 50 000. This value is used by the examples involving latent variables, where the likelihood contains an integral over the distribution of the latent construct. Defining the number of draws in a separate file makes the examples easier to maintain and ensures that all models relying on simulation use the same level of numerical accuracy.

B.4 one_latent_variable_spec.py

```
"""
Latent variables (SPEC ONLY)

Pure specification: what depends on what.

```

```
Michel Bierlaire
:Wed Mar 04 2026, 13:48:57
"""

from biogeme.latent_variables import (
    LatentVariable,
    PositiveParameterSpec,
    StructuralEquation,
)

DEFAULT_SIGMA_START = 10.0

car_centric_attitude = LatentVariable(
    name='car_centric_attitude',
    structural_equation=StructuralEquation(
        name='car_centric_attitude',
        intercept=True,
        explanatory_variables=[
            'top_manager',
            'car_oriented_parents',
            'high_education',
            'low_education',
            'used_to_go_to_school_by_car',
        ],
    ),
    indicators={
        'Envir01',
        'Envir02',
        'Envir06',
        'Mobil03',
        'Mobil05',
        'Mobil08',
        'Mobil09',
        'Mobil10',
        'LifSty07',
    },
    structural_sigma=PositiveParameterSpec(start=DEFAULT_SIGMA_START),
)

latent_variables = [car_centric_attitude]
```

This file defines the semantic specification of the latent variable used throughout the examples. It does not build a likelihood function and does not estimate any model. Instead, it describes the structure of the latent construct independently of the measurement and choice models in which it will later be used.

The object car_centric_attitude is declared as a LatentVariable. Its structural equation is specified by a StructuralEquation object with an intercept and five observed explanatory variables: top_manager, car_oriented_parents, high_education, low_education, and used_to_go_to_school_by_car. These variables correspond to the causes of the latent attitude in the MIMIC terminology introduced earlier.

The file also lists the psychometric indicators associated with the latent variable. These indicators are not modeled here; only their names are attached to the semantic specification. The actual measurement equations are introduced later, depending on whether the indicators are treated as continuous, ordered logit, or ordered probit responses.

Finally, the standard deviation of the structural disturbance is declared using a PositiveParameterSpec, ensuring that the corresponding parameter is constrained to be positive during estimation. The file ends by collecting the latent variable in the list latent_variables, which provides a common interface for the scripts that construct the different MIMIC and hybrid choice models.

B.5 optima.py

```
"""
.. _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
"""

```

```python
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)
    # 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})
    df.drop(df[df["number_of_cars"] == -1].index, inplace=True)

    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')
```

The file optima.py is responsible for preparing the data used throughout the examples. It reads the Optima case-study data from the file optima.dat, applies a number of filters, creates derived variables, and returns a Biogeme Database object.

The function read_data first imports the data using pandas. Observations that are not compatible with the modeling assumptions are removed: invalid choices, non-workers, tours with only one trip, observations with zero travel times or zero distance, and observations where the car alternative is chosen despite being unavailable. This cleaning step defines the estimation sample used in the subsequent examples.

The function then creates several variables used in the choice and latent-variable specifications. These include availability indicators, normalized sampling weights, socio-demographic categories, education indicators, car-availability indicators, trip-purpose indicators, and scaled versions of travel time, cost, and distance. The scaling of continuous variables is purely numerical: it improves the conditioning of the optimization problem without changing the behavioral content of the model.

Note that the normalized sampling weight is prepared for reuse by other case studies, but it is not supplied as a Biogeme weight in the present examples; the reported estimations are therefore unweighted.

Finally, the lower part of the file defines a large collection of Biogeme Variable objects. These objects provide symbolic access to the columns of the database and to the variables created inside read_data. Other modules can therefore import the variables directly from optima.py, which keeps the model specifications concise and separates data preparation from the definition of utilities, structural equations, and measurement equations.

B.6 plot_h01_mode_logit.py

```
"""
Baseline mode choice model: maximum likelihood estimation
=========================================================

This example estimates a multinomial logit model for transportation mode
choice using maximum likelihood and Biogeme.

It is the baseline specification of the hybrid-choice tutorial. The model
contains only observed choice variables and standard utility functions: there
are no latent variables, structural equations, or measurement equations. The
results therefore provide a reference against which the hybrid-choice
specifications introduced later can be compared.

```

```
The script performs the following steps:

- load the Optima mode-choice data,
- build the utility functions from the common tutorial specification,
- construct the logit log-likelihood,
- estimate the model, or reload previously saved estimation results,
- display the estimated parameters as pandas and LaTeX tables.

Michel Bierlaire
Sat Jun 06 2026, 15:23:41
"""

from choice_latent_variables import generate_availability, generate_utility_functions
from optima import (
    Choice,
    read_data,
)

import biogeme.biogeme_logging as blog
from biogeme.biogeme import BIOGEME
from biogeme.models import loglogit
from biogeme.results_processing import (
    get_latex_estimated_parameters,
    get_latex_general_statistics,
    get_pandas_estimated_parameters,
)

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

# %%
# Load the Optima mode-choice data.
database = read_data()

# Build the utility functions for the transportation alternatives.
utilities = generate_utility_functions()

# Availability
availability = generate_availability()

# Construct the log-likelihood of the multinomial logit model.
log_likelihood = loglogit(utilities, availability, Choice)

# Create the Biogeme object used for estimation.
biogeme = BIOGEME(
    database,
    log_likelihood,
)
biogeme.model_name = 'plot_h01_mode_logit'

# Estimate the model, or reload the saved results if they are already available.
yaml_file_name = f'saved_results/{biogeme.model_name}.yaml'
results = biogeme.estimate_or_load(yaml_file_name=yaml_file_name)

# Display a compact summary and the estimated parameters.
print(results.short_summary())
print(get_pandas_estimated_parameters(estimation_results=results))

general_statistics = get_latex_general_statistics(estimation_results=results)
print(general_statistics)

estimated_parameters = get_latex_estimated_parameters(estimation_results=results)
print(estimated_parameters[''])
```

This script implements the baseline model introduced in Section 4.2. It contains only the choice component of the model: no latent variable, structural equation, or measurement equation is involved. The script first reads the Optima data set using the common data-loading function defined in `optima.py`. It then obtains the systematic utilities from `generate_utility_functions`, so that the utility specification is shared with the hybrid models introduced later. The log-likelihood is constructed with Biogeme's `loglogit` function, using the observed choice variable `Choice`.

The resulting expression is passed to a `BIOGEME` object, which performs maximum-likelihood estimation. The call to `estimate_or_load` either estimates the model or reloads previously saved results from the corresponding YAML file. Finally, the script prints a short summary of the estimation results and generates both pandas and LaTeX tables of the estimated parameters. This output provides the reference results against which the latent-variable and hybrid-choice specifications are compared in the following examples.

The function `estimate_or_load` first checks whether the specified YAML result file already exists. If it exists, the saved results are loaded and the model is not re-estimated. Consequently, changing the source code, the number of draws, or the estimation settings does not automatically invalidate an existing result file.

To force a new estimation, rename or remove the corresponding YAML file and, when present, the associated iteration file. An alternative is to use a separate output directory or model name for each specification. This is particularly important when comparing alternative implementations.

B.7 plot_h02_lv_mimic_gauss.py

```python
"""
Gaussian MIMIC model: maximum likelihood estimation
==================================================

This example estimates a Gaussian MIMIC model with one latent variable and no
choice component. The observed Likert indicators are treated as continuous
responses and modeled with Gaussian measurement equations.

The example introduces the latent-variable-only setting used later in the
hybrid-choice tutorial. The latent-variable structure itself is imported from a
separate semantic specification file. This script adds the Gaussian measurement
configuration, the normalization constraints required for identification, and
the maximum-likelihood estimation setup.

The script performs the following steps:

- load the latent-variable and indicator specifications,
- define a Gaussian measurement configuration for all indicators,
- define the normalization constraints used for identification,
- resolve the semantic specification into an estimable model,
- build the Biogeme expressions from the resolved model,
- integrate over the latent variable using Monte Carlo draws,
- construct the corresponding log-likelihood,
- estimate the model, or reload previously saved estimation results,
- display the estimated parameters as pandas and LaTeX tables.

Michel Bierlaire
Mon Jun 15 2026, 09:54:37
"""

from __future__ import annotations

from likert_spec import likert_indicators, likert_types
from number_of_draws import NUMBER_OF_DRAWS
from one_latent_variable_spec import DEFAULT_SIGMA_START, latent_variables
from optima import read_data

import biogeme.biogeme_logging as blog
from biogeme.biogeme import BIOGEME
from biogeme.expressions import log
from biogeme.latent_variables import (
    BuildContext,
    EstimationMode,
    Fixing,
    IndicatorMeasurementSpec,
    MeasurementConfiguration,
    MeasurementIntercept,
    MeasurementLoading,
    MeasurementModel,
    NormalizationPlan,
    PositiveParameterSpec,
    PositivityMode,
    build_biogeme_model,
    resolve_model,
)
from biogeme.results_processing import (
    get_latex_estimated_parameters,
    get_latex_general_statistics,
    get_pandas_estimated_parameters,
)

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

# %%
# Gaussian measurement configuration for all Likert indicators.
measurement_configuration = MeasurementConfiguration(
    specifications=[
        IndicatorMeasurementSpec(
            indicator_name=indicator.name,
            measurement_model=MeasurementModel.GAUSSIAN,
            measurement_sigma=PositiveParameterSpec(start=DEFAULT_SIGMA_START),
        )
        for indicator in likert_indicators
    ]
)

# %%
# Load the Optima data.
# ---------------------
database = read_data()

# %%
# Define the build context for maximum likelihood estimation.
# -----------------------------------------------------------
# The build context specifies how the semantic latent-variable specification is
# translated into estimable Biogeme expressions. We start from the default
# maximum-likelihood context and explicitly select the log-exp parameterization
# for positive parameters.
default_context = BuildContext.default(EstimationMode.MAXIMUM_LIKELIHOOD)
context = BuildContext(
    estimation_mode=default_context.estimation_mode,
    draw_type=default_context.draw_type,
    positivity_mode=PositivityMode.LOG_EXP,
    naming=default_context.naming,
    ordinal_eps=default_context.ordinal_eps,
    ordinal_enforce_order=default_context.ordinal_enforce_order,
)

# %%
# Identification constraints for the latent variable
# --------------------------------------------------
# A latent variable is not directly observed, so its location and scale are
# not identified unless we impose normalization constraints.
#
# We use a reference-indicator strategy based on indicator ``Envir01``:
```

```
#
# 1. The intercept of the reference indicator is fixed to 0.
#    This anchors the location of the latent-variable scale.
#
# 2. The loading of the reference indicator on the latent variable is fixed
#    to -1.
#    This anchors the scale of the latent variable and fixes its orientation.
#    The negative sign means that higher values of the latent variable imply
#    lower expected values for ``Envir01``.
#
# With these two constraints in place, the remaining intercepts and loadings
# are interpreted relative to the reference indicator.
normalization_plan = NormalizationPlan()

normalization_plan.add(
    Fixing(MeasurementIntercept('Envir01'), 0.0, note='reference indicator: location')
)
normalization_plan.add(
    Fixing(
        MeasurementLoading('car_centric_attitude', 'Envir01'),
        -1.0,
        note='reference indicator: scale and orientation',
    )
)

# %%
# Resolve the semantic specification.
# -----------------------------------
# The resolver combines the latent-variable specification, the indicator
# definitions, the Gaussian measurement configuration, and the normalization plan
# into an internal resolved model.
resolved_model = resolve_model(
    latent_variables=latent_variables,
    likert_indicators=likert_indicators,
    likert_types=likert_types,
    measurement_configuration=measurement_configuration,
    context=context,
    normalization_plan=normalization_plan,
)

# %%
# Build the Biogeme expressions.
# ------------------------------
# The builder translates the resolved model into expressions that can be used by
# Biogeme for maximum-likelihood estimation.
built_model = build_biogeme_model(resolved_model)

# %%
# Log-likelihood
# --------------
# The builder provides the Monte-Carlo integrated measurement likelihood.
# Taking its logarithm yields the log-likelihood used for maximum-likelihood
# estimation.
integrated_likelihood = built_model.integrated_likelihood
log_likelihood = log(integrated_likelihood)

# %%
# Estimate the model with Biogeme.
# --------------------------------
# Existing results are reloaded from the YAML file when available.
biogeme = BIOGEME(
    database,
    log_likelihood,
    number_of_draws=NUMBER_OF_DRAWS,
    calculating_second_derivatives='never',
    numerically_safe=False,
    max_iterations=5_000,
    group_of_parameters=built_model.parameter_groups,
)
biogeme.model_name = 'plot_h02_lv_mimic_gauss'

yaml_file_name = f'saved_results/{biogeme.model_name}.yaml'
results = biogeme.estimate_or_load(yaml_file_name=yaml_file_name)

# %%
# Display a compact summary and the estimated parameters.
print(results.short_summary())
pandas_results = get_pandas_estimated_parameters(
    estimation_results=results,
    group_of_parameters=built_model.parameter_groups,
)
for group_name, pandas_table in pandas_results.items():
    print(group_name if group_name else 'Estimated parameters')
    print(pandas_table)

general_statistics = get_latex_general_statistics(estimation_results=results)
print(general_statistics)

estimated_parameters = get_latex_estimated_parameters(
    estimation_results=results,
    group_of_parameters=built_model.parameter_groups,
)
for group_name, latex_table in estimated_parameters.items():
    print(group_name if group_name else 'Estimated parameters')
    print(latex_table)
```

The file plot_h02_lv_mimic_gauss.py implements the Gaussian MIMIC model discussed in Section 4.3. It is the first example where the latent-variable machinery is used explicitly, but it does not yet contain a choice component. The purpose of the script is therefore to estimate the structural equation of the latent variable together with continuous measurement equations for the Likert indicators.

The script first imports the semantic specification of the latent variable from one_latent_variable_spec .py, the list of Likert indicators from likert_spec .py, and the Optima data from optima.py. It then defines a MeasurementConfiguration in which each Likert indicator is associated with a Gaussian measurement model. In this specification, the indicators are treated as continuous responses, and each measurement equation has its own positive standard deviation parameter.

The next part of the script defines the build context used for maximum likelihood estimation. Positive parameters are represented through a log-exponential parametrization, which guarantees positivity while leaving the optimizer unconstrained in the transformed parameter. The normalization plan then imposes the identification constraints for the latent variable. The indicator Envir01 is used as the reference indicator: its intercept is fixed to zero, and its loading on the car-centric latent variable is fixed to -1. These two constraints anchor respectively the location, the scale, and the orientation of the latent-variable scale.

The semantic model is then resolved by combining the latent-variable specification, the indicator definitions, the Gaussian measurement configuration, and the normalization plan. The resolved model is translated into Biogeme expressions by build_biogeme_model. In particular, the builder provides the measurement likelihood integrated over the latent variable using Monte-Carlo simulation. Taking the logarithm of this integrated likelihood gives the log-likelihood maximized by Biogeme.

Finally, the script creates the BIOGEME object, specifies the number of simulation draws, estimates the model or reloads saved results, and prints the output both as pandas tables and as LaTeX tables. The parameter groups generated by the resolved model are used to organize the output according to the structural equation and the measurement equations.

B.8 plot_h03_mode_lv_gauss_seq.py

```python
"""
Sequential estimation of a choice model with a latent variable
==============================================================

This example illustrates the second step of a sequential estimation
procedure.

The latent variable and its structural equation are first estimated using the
Gaussian MIMIC model of ``plot_h02_lv_mimic_gauss``. The estimated structural
parameters are then loaded from the saved results file and used to construct
the latent-variable expression appearing in the choice model.

Only the choice-model parameters are estimated in this script. The latent
variable is not re-estimated jointly with the measurement equations.

Michel Bierlaire
Mon Jun 15 2026, 09:54:37
"""

import sys

from choice_latent_variables import generate_utility_functions
from number_of_draws import NUMBER_OF_DRAWS
from one_latent_variable_spec import car_centric_attitude
from optima import (
    Choice,
    read_data,
)

import biogeme.biogeme_logging as blog
from biogeme.biogeme import BIOGEME
from biogeme.expressions import (
    Draws,
    MonteCarlo,
    MultipleSum,
    Variable,
    exp,
    log,
)
from biogeme.models import logit
from biogeme.results_processing import (
    EstimationResults,
    get_latex_estimated_parameters,
    get_latex_general_statistics,
    get_pandas_estimated_parameters,
)

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

# %%
# Load data
database = read_data()

# %%
# Read the results of the previously estimated MIMIC model.
FILE_WITH_ESTIMATION_RESULTS = 'saved_results/plot_h02_lv_mimic_gauss.yaml'
try:
    estimation_results = EstimationResults.from_yaml_file(
```

```python
        filename=FILE_WITH_ESTIMATION_RESULTS
    )
except FileNotFoundError:
    print(
        f'File {FILE_WITH_ESTIMATION_RESULTS} does not exist. Sequential estimation cannot be performed.'
    )
    sys.exit()
estimated_parameters: dict[str, float] = estimation_results.get_beta_values()

# %%
# Reconstruct the latent-variable expression using the parameter estimates
# obtained from the MIMIC model. The sequential specification keeps both the
# deterministic component and the random structural disturbance.
list_of_explanatory_variables = (
    car_centric_attitude.structural_equation.explanatory_variables
)

# The sigma parameter was estimated on the log scale in the MIMIC model.
# We therefore exponentiate it before scaling the structural disturbance.
terms = [
    estimated_parameters[f'struct_car_centric_attitude_{var}'] * Variable(var)
    for var in list_of_explanatory_variables
]

structural_sigma = exp(estimated_parameters['struct_car_centric_attitude_sigma_log'])
structural_error = structural_sigma * Draws(
    'car_centric_attitude_draw', 'NORMAL_MLHS_ANTI'
)

car_centric_attitude = MultipleSum(terms) + structural_error
latent_expressions = {'car_centric_attitude': car_centric_attitude}

# %%
# Utility functions including the latent-variable effect.
v = generate_utility_functions(latent_expressions=latent_expressions)

# %%
# Choice model estimated conditionally on the latent-variable specification.
conditional_likelihood = logit(v, None, Choice)

loglikelihood = log(MonteCarlo(conditional_likelihood))

biogeme = BIOGEME(
    database,
    loglikelihood,
    number_of_draws=NUMBER_OF_DRAWS,
    numerically_safe=True,
    calculating_second_derivatives='never',
    max_iterations=5_000,
)
biogeme.model_name = 'plot_h03_model_lv_gauss_seq'

yaml_file_name = f'saved_results/{biogeme.model_name}.yaml'
results = biogeme.estimate_or_load(yaml_file_name=yaml_file_name)

print(results.short_summary())
print(get_pandas_estimated_parameters(estimation_results=results))

general_statistics = get_latex_general_statistics(estimation_results=results)
print(general_statistics)

estimated_parameters = get_latex_estimated_parameters(estimation_results=results)
for group_name, latex_table in estimated_parameters.items():
    print(group_name if group_name else 'Estimated parameters')
    print(latex_table)
```

The script plot_h03_mode_lv_gauss_seq.py implements the sequential estimation approach discussed in Section 4.4. It assumes that the latent-variable model has already been estimated by plot_h02_lv_mimic_gauss.py, and starts by loading the corresponding estimation results from the saved YAML file. If this file is not available, the script stops, since the sequential estimation cannot be performed without the first-step estimates.

The next part of the script reconstructs the car-centric latent variable using the estimated structural-equation parameters. The explanatory variables are taken directly from the semantic specification of the latent variable, ensuring consistency with the MIMIC model. The estimated standard deviation of the structural disturbance is stored on the log scale and is therefore exponentiated before being multiplied by a normal draw. The resulting expression contains both the deterministic component of the structural equation and the random structural error term.

The reconstructed latent variable is then passed to generate_utility_functions, which inserts it into the utility of the car alternative. The choice probability is evaluated using the logit model, conditionally on the simulated value of the latent variable. Because the latent variable remains random, the conditional likelihood is integrated by Monte-Carlo simulation. The script then creates the BIOGEME object, estimates or loads the model, and prints the usual summary statistics and estimated parameters in both pandas and LaTeX formats.

This script therefore illustrates the main feature of sequential hybrid choice estimation: the measurement model is not part of the second-step likelihood. The indicators influence the choice model only through the latent-variable parameters estimated previously, which are treated as fixed in this script.

B.9 plot_h04_mode_lv_gauss_simult.py

```python
"""
Gaussian hybrid mode choice model: simultaneous maximum likelihood estimation
=============================================================================

This example estimates a hybrid mode choice model with one latent variable and
Gaussian measurement equations. The observed Likert indicators are treated as
continuous responses and modeled with Gaussian measurement equations. Responses
using the type's neutral or missing labels (6 and -1 in this case) contribute a
neutral factor to the measurement likelihood and are therefore excluded from
the corresponding Gaussian measurement equation.

In contrast with the sequential specification introduced previously, the
measurement component and the mode-choice component are estimated jointly. The
conditional likelihood of the indicators and the conditional likelihood of the
observed choice are multiplied conditionally on the latent variable, and the
resulting expression is integrated over the latent-variable distribution.

The latent-variable structure is imported from a separate semantic specification
file. This script adds the Gaussian measurement configuration, the normalization
constraints required for identification, the mode-choice utilities, and the
simultaneous maximum-likelihood estimation setup.

The script performs the following steps:

- load the latent-variable and indicator specifications,
- define a Gaussian measurement configuration for all indicators,
- define the normalization constraints used for identification,
- resolve the semantic specification into an estimable model,
- build the Biogeme expressions from the resolved model,
- build the choice utilities, including the latent-variable term,
- combine the measurement and choice conditional likelihoods,
- integrate the combined conditional likelihood over the latent variable,
- estimate the hybrid model, or reload previously saved estimation results,
- display the estimated parameters as grouped pandas and LaTeX tables.

Michel Bierlaire
Mon Jun 15 2026, 09:54:37
"""

from __future__ import annotations

from choice_latent_variables import generate_availability, generate_utility_functions
from likert_spec import likert_indicators, likert_types
from number_of_draws import NUMBER_OF_DRAWS
from one_latent_variable_spec import latent_variables
from optima import Choice, read_data

import biogeme.biogeme_logging as blog
from biogeme.biogeme import BIOGEME
from biogeme.expressions import MonteCarlo, log
from biogeme.latent_variables import (
    BuildContext,
    BuiltBiogemeModel,
    EstimationMode,
    Fixing,
    IndicatorMeasurementSpec,
    MeasurementConfiguration,
    MeasurementIntercept,
    MeasurementLoading,
    MeasurementModel,
    NormalizationPlan,
    PositiveParameterSpec,
    PositivityMode,
    build_biogeme_model,
    resolve_model,
)
from biogeme.models import logit
from biogeme.results_processing import (
    get_latex_estimated_parameters,
    get_latex_general_statistics,
    get_pandas_estimated_parameters,
)

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

DEFAULT_MEASUREMENT_SIGMA_START = 10.0

# %%
# Gaussian measurement configuration for all Likert indicators.
# ------------------------------------------------------------
# The neutral-label semantics are supplied by ``likert_types`` and are applied
# by the latent-variable builder to these Gaussian measurement equations.
measurement_configuration = MeasurementConfiguration(
    specifications=[
        IndicatorMeasurementSpec(
            indicator_name=indicator.name,
            measurement_model=MeasurementModel.GAUSSIAN,
            measurement_sigma=PositiveParameterSpec(
                start=DEFAULT_MEASUREMENT_SIGMA_START
            ),
        )
        for indicator in likert_indicators
    ]
)

# %%
# Load the Optima data.
# ---------------------
database = read_data()
```

```python
# %%
# Define the build context for maximum likelihood estimation.
# -----------------------------------------------------------
# The build context specifies how the semantic latent-variable specification is
# translated into estimable Biogeme expressions. We start from the default
# maximum-likelihood context and explicitly select the log-exp parameterization
# for positive parameters.
default_context = BuildContext.default(EstimationMode.MAXIMUM_LIKELIHOOD)
context = BuildContext(
    estimation_mode=default_context.estimation_mode,
    draw_type=default_context.draw_type,
    positivity_mode=PositivityMode.LOG_EXP,
    naming=default_context.naming,
    ordinal_eps=default_context.ordinal_eps,
    ordinal_enforce_order=default_context.ordinal_enforce_order,
)

# %%
# Identification constraints for the latent variable
# --------------------------------------------------
# A latent variable is not directly observed, so its location and scale are
# not identified unless normalization constraints are imposed.
#
# We use a reference-indicator strategy based on ``Envir01``:
#
# - the intercept of ``Envir01`` is fixed to 0.0, which anchors the location
#   of the latent-variable scale;
# - the loading of ``Envir01`` on ``car_centric_attitude`` is fixed to -1.0,
#   which anchors the scale and fixes the orientation of the latent variable.
#
# The negative sign means that higher values of ``car_centric_attitude`` imply
# lower expected values for ``Envir01``. With these constraints in place, the
# remaining intercepts and loadings are interpreted relative to the reference
# indicator.
normalization_plan = NormalizationPlan()

normalization_plan.add(
    Fixing(
        MeasurementIntercept('Envir01'),
        0.0,
        note='car_centric_attitude reference indicator: location',
    )
)
normalization_plan.add(
    Fixing(
        MeasurementLoading('car_centric_attitude', 'Envir01'),
        -1.0,
        note='car_centric_attitude reference indicator: scale and orientation',
    )
)

# %%
# Resolve the semantic specification.
# -----------------------------------
# The resolver combines the latent-variable specification, the indicator
# definitions, the Gaussian measurement configuration, and the normalization plan
# into an internal resolved model.
resolved_model = resolve_model(
    latent_variables=latent_variables,
    likert_indicators=likert_indicators,
    likert_types=likert_types,
    measurement_configuration=measurement_configuration,
    context=context,
    normalization_plan=normalization_plan,
)

# %%
# Build the Biogeme expressions.
# ------------------------------
# The builder translates the resolved model into expressions that can be used by
# Biogeme for simultaneous maximum-likelihood estimation. It also provides
# report-ready parameter groups based on the parameters that are actually
# estimated.
built_model: BuiltBiogemeModel = build_biogeme_model(resolved_model)

# %%
# Choice utilities including the latent-variable term.
# ----------------------------------------------------
utilities = generate_utility_functions(built_model.latent_expressions)

# %%
# Conditional likelihood of the mode choice model
# -----------------------------------------------
availability = generate_availability()
conditional_choice_likelihood = logit(utilities, availability, Choice)

# %%
# Combined conditional likelihood
# -------------------------------
# The measurement component and the mode-choice component are combined
# conditionally on the latent variable. This is the key difference with the
# sequential approach: both parts are estimated jointly in a single likelihood.
combined_conditional_likelihood = (
    built_model.conditional_likelihood * conditional_choice_likelihood
)

# %%
# Log-likelihood
# --------------
# The combined conditional likelihood is integrated over the latent variable
# by Monte Carlo. Taking the logarithm of the resulting integrated likelihood
# yields the log-likelihood used for simultaneous maximum-likelihood
# estimation.
integrated_likelihood = MonteCarlo(combined_conditional_likelihood)
log_likelihood = log(integrated_likelihood)

# %%
```

```python
# Estimate the model with Biogeme.
# ----------------------------------------
# Existing results are reloaded from the YAML file when available.
biogeme = BIOGEME(
    database,
    log_likelihood,
    number_of_draws=NUMBER_OF_DRAWS,
    calculating_second_derivatives='never',
    max_iterations=5_000,
    group_of_parameters=built_model.parameter_groups,
)
biogeme.model_name = 'plot_h04_mode_lv_gauss_simult'

yaml_file_name = f'saved_results/{biogeme.model_name}.yaml'
results = biogeme.estimate_or_load(yaml_file_name=yaml_file_name)

# %%
# Display a compact summary and the estimated parameters.
print(results.short_summary())
pandas_results = get_pandas_estimated_parameters(
    estimation_results=results,
    group_of_parameters=built_model.parameter_groups,
)
for group_name, pandas_table in pandas_results.items():
    print(group_name if group_name else 'Estimated parameters')
    print(pandas_table)

general_statistics = get_latex_general_statistics(estimation_results=results)
print(general_statistics)

estimated_parameters = get_latex_estimated_parameters(
    estimation_results=results,
    group_of_parameters=built_model.parameter_groups,
)
for group_name, latex_table in estimated_parameters.items():
    print(group_name if group_name else 'Estimated parameters')
    print(latex_table)
```

The file plot_h04_mode_lv_gauss_simult.py implements the first simultaneous hybrid choice model of the tutorial. It combines the mode choice model with the Gaussian MIMIC measurement model in a single likelihood function. The script first defines Gaussian measurement equations for all Likert indicators, using positive scale parameters for the measurement errors. It then loads the Optima data and creates the build context used to translate the semantic latent-variable specification into Biogeme expressions.

A central part of the file is the normalization plan. As in the MIMIC model, the indicator Envir01 is used as reference indicator: its intercept is fixed to zero, and its loading on the car-centric latent variable is fixed to -1. These two constraints fix the location, scale, and orientation of the latent variable. The semantic model is then resolved and converted into Biogeme expressions. The utility functions are generated with the latent variable included in the car utility, and the logit choice probability is constructed.

The main difference with the sequential specification is the construction of the likelihood. The conditional likelihood of the measurement model and the conditional likelihood of the mode choice model are multiplied conditionally on the latent variable. This combined likelihood is then integrated over the latent-variable distribution using Monte-Carlo simulation. The resulting expression is the log-likelihood of the simultaneous hybrid choice model. The last part of the script estimates the model, or reloads previously saved results, and prints the estimated parameters grouped according to their role in the model.

B.10 plot_h05_mode_lv_ordlogit_simult.py

```python
"""
Ordered-logit hybrid mode choice model: simultaneous maximum likelihood estimation
==================================================================================

This example estimates a hybrid mode choice model with one latent variable and
ordered-logit measurement equations. The observed Likert indicators are modeled
as ordinal responses, and the associated threshold parameters are estimated
jointly with the latent-variable and choice-model parameters.

Compared with the previous Gaussian specification, only the measurement model
changes: the hybrid structure, the latent variable, and the simultaneous
maximum-likelihood estimation strategy remain the same.

The latent-variable structure is imported from a separate semantic specification
file. This script adds the ordered-logit measurement configuration, the
normalization constraints required for identification, the mode-choice utilities,
and the simultaneous maximum-likelihood estimation setup.

The script performs the following steps:

- load the latent-variable and indicator specifications,
- define an ordered-logit measurement configuration for all indicators,
- define the normalization constraints used for identification,
- resolve the semantic specification into an estimable model,
- build the Biogeme expressions from the resolved model,
- build the choice utilities, including the latent-variable term,
```

```
- combine the ordered-logit measurement and choice conditional likelihoods,
- integrate the combined conditional likelihood over the latent variable,
- estimate the hybrid model, or reload previously saved estimation results,
- display the estimated parameters as grouped pandas and LaTeX tables.

Michel Bierlaire
Mon Jun 15 2026, 09:54:37
"""

from __future__ import annotations

from choice_latent_variables import generate_availability, generate_utility_functions
from likert_spec import likert_indicators, likert_types
from number_of_draws import NUMBER_OF_DRAWS
from one_latent_variable_spec import latent_variables
from optima import Choice, read_data

import biogeme.biogeme_logging as blog
from biogeme.biogeme import BIOGEME
from biogeme.expressions import MonteCarlo, log
from biogeme.latent_variables import (
    BuildContext,
    EstimationMode,
    Fixing,
    IndicatorMeasurementSpec,
    MeasurementConfiguration,
    MeasurementIntercept,
    MeasurementLoading,
    MeasurementModel,
    MeasurementSigma,
    NormalizationPlan,
    PositiveParameterSpec,
    PositivityMode,
    build_biogeme_model,
    resolve_model,
)
from biogeme.models import logit
from biogeme.results_processing import (
    get_latex_estimated_parameters,
    get_latex_general_statistics,
    get_pandas_estimated_parameters,
)

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

DEFAULT_MEASUREMENT_SIGMA_START = 10.0


# %%
# Ordered-logit measurement configuration for all Likert indicators.
# -------------------------------------------------------------------
measurement_configuration = MeasurementConfiguration(
    specifications=[
        IndicatorMeasurementSpec(
            indicator_name=indicator.name,
            measurement_model=MeasurementModel.ORDERED_LOGIT,
            measurement_sigma=PositiveParameterSpec(
                start=DEFAULT_MEASUREMENT_SIGMA_START
            ),
        )
        for indicator in likert_indicators
    ]
)

# %%
# Load the Optima data.
# ---------------------
database = read_data()

# %%
# Define the build context for maximum likelihood estimation.
# -----------------------------------------------------------
# The build context specifies how the semantic latent-variable specification is
# translated into estimable Biogeme expressions. We start from the default
# maximum-likelihood context and explicitly select the log-exp parameterization
# for positive parameters. The ordinal options are relevant here because the
# measurement equations use ordered-logit probabilities.
default_context = BuildContext.default(EstimationMode.MAXIMUM_LIKELIHOOD)
context = BuildContext(
    estimation_mode=default_context.estimation_mode,
    draw_type=default_context.draw_type,
    positivity_mode=PositivityMode.LOG_EXP,
    naming=default_context.naming,
    ordinal_eps=default_context.ordinal_eps,
    ordinal_enforce_order=default_context.ordinal_enforce_order,
)

# %%
# Identification constraints for the latent variable
# --------------------------------------------------
# Two layers of normalization are needed in this ordered-logit specification.
#
# 1. Ordered-logit measurement-scale normalization
#    The latent response underlying the ordinal indicators has an arbitrary
#    scale. We therefore fix one measurement sigma to 1.0 in order to anchor
#    the overall scale of the ordered-logit measurement model.
#
# 2. Latent-variable normalization by reference indicator
#    The latent variable itself is not directly observed, so its location and
#    orientation must also be fixed. We use indicator ``Envir01`` as the
#    reference indicator:
#
#    - its intercept is fixed to 0.0, which anchors the location of the latent
#      variable;
#    - its loading on ``car_centric_attitude`` is fixed to -1.0, which anchors
#      the scale and fixes the orientation of the latent variable.
#
```

```
# The negative sign means that higher values of ``car_centric_attitude`` imply
# lower expected values for ``Envir01``. With these constraints in place, the
# remaining intercepts, loadings, and thresholds are interpreted relative to
# the chosen reference and measurement scale.
normalization_plan = NormalizationPlan()

normalization_plan.add(
    Fixing(
        MeasurementSigma('Envir01'),
        1.0,
        note='ordered-logit measurement model: scale normalization',
    )
)

normalization_plan.add(
    Fixing(
        MeasurementIntercept('Envir01'),
        0.0,
        note='car_centric_attitude reference indicator: location',
    )
)
normalization_plan.add(
    Fixing(
        MeasurementLoading('car_centric_attitude', 'Envir01'),
        -1.0,
        note='car_centric_attitude reference indicator: scale and orientation',
    )
)

# %%
# Resolve the semantic specification.
# -----------------------------------
# The resolver combines the latent-variable specification, the indicator
# definitions, the ordered-logit measurement configuration, and the normalization
# plan into an internal resolved model.
resolved_model = resolve_model(
    latent_variables=latent_variables,
    likert_indicators=likert_indicators,
    likert_types=likert_types,
    measurement_configuration=measurement_configuration,
    context=context,
    normalization_plan=normalization_plan,
)

# %%
# Build the Biogeme expressions.
# ------------------------------
# The builder translates the resolved model into expressions that can be used by
# Biogeme for simultaneous maximum-likelihood estimation. It also provides
# report-ready parameter groups based on the parameters that are actually
# estimated.
built_model = build_biogeme_model(resolved_model)

# %%
# Choice utilities including the latent-variable term.
# ----------------------------------------------------
utilities = generate_utility_functions(built_model.latent_expressions)

# %%
# Conditional likelihood of the mode choice model
# -----------------------------------------------

availability = generate_availability()
conditional_choice_likelihood = logit(utilities, availability, Choice)

# %%
# Combined conditional likelihood
# -------------------------------
# The ordered-logit measurement component and the mode-choice component are
# combined conditionally on the latent variable. As in the previous step,
# both parts are estimated jointly in a single likelihood.
combined_conditional_likelihood = (
    built_model.conditional_likelihood * conditional_choice_likelihood
)

# %%
# Log-likelihood
# --------------
# The combined conditional likelihood is integrated over the latent variable
# by Monte Carlo. Taking the logarithm of the resulting integrated likelihood
# yields the log-likelihood used for simultaneous maximum-likelihood
# estimation.
integrated_likelihood = MonteCarlo(combined_conditional_likelihood)
log_likelihood = log(integrated_likelihood)


# %%
# Estimate the model with Biogeme.
# --------------------------------
# Existing results are reloaded from the YAML file when available.
biogeme = BIOGEME(
    database,
    log_likelihood,
    number_of_draws=NUMBER_OF_DRAWS,
    calculating_second_derivatives='never',
    max_iterations=5_000,
    group_of_parameters=built_model.parameter_groups,
)
biogeme.model_name = 'plot_h05_mode_lv_ordlogit_simult'

yaml_file_name = f'saved_results/{biogeme.model_name}.yaml'
results = biogeme.estimate_or_load(yaml_file_name=yaml_file_name)

# %%
# Display a compact summary and the estimated parameters.
print(results.short_summary())
pandas_results = get_pandas_estimated_parameters(
```

```
    estimation_results=results,
    group_of_parameters=built_model.parameter_groups,
)
for group_name, pandas_table in pandas_results.items():
    print(group_name if group_name else 'Estimated parameters')
    print(pandas_table)

general_statistics = get_latex_general_statistics(estimation_results=results)
print(general_statistics)

estimated_parameters = get_latex_estimated_parameters(
    estimation_results=results,
    group_of_parameters=built_model.parameter_groups,
)
for group_name, latex_table in estimated_parameters.items():
    print(group_name if group_name else 'Estimated parameters')
    print(latex_table)
```

The file plot_h05_mode_lv_ordlogit_simult.py implements the simultaneous hybrid choice model with ordered-logit measurement equations. It starts from the same semantic specification of the car-centric latent variable used in the previous examples, and only changes the way in which the Likert indicators are modeled. Instead of treating them as continuous responses, the script declares an ordered-logit measurement model for each indicator through a MeasurementConfiguration.

The code is organized according to the theoretical decomposition of the hybrid choice model. First, it loads the data and defines the estimation context, including the log-exp parameterization used to enforce positivity constraints. It then specifies the normalization plan. In the ordinal case, two types of normalization are needed: one measurement scale parameter is fixed in order to identify the scale of the ordered-logit measurement layer, and the reference indicator Envir01 is used to fix the location, scale, and orientation of the latent variable. Its intercept is fixed to zero and its loading on the car-centric attitude is fixed to -1.

The semantic latent-variable specification, the Likert indicators, the ordered-logit measurement configuration, and the normalization plan are then resolved into an estimable model. The resulting Biogeme expressions contain the conditional likelihood of the measurement model, the latent-variable expression, and the corresponding parameter groups used later to organize the output tables.

The mode-choice component is built separately by calling generate_utility_functions, which inserts the latent variable into the utility of the car alternative. The standard logit choice probability is then multiplied by the ordered-logit measurement likelihood. This product is conditional on the latent variable. The final likelihood is obtained by Monte-Carlo integration over the latent variable, followed by the logarithm of the integrated likelihood.

Finally, the script creates the BIOGEME object, estimates the model or reloads previously saved results, and prints the output in several formats. The use of parameter groups makes it possible to separate the reported estimates into structural parameters, measurement parameters, threshold parameters, and choice-model parameters, consistently with the theoretical structure of the hybrid choice model.

B.11 plot_h06_mode_lv_ordprobit_simult.py

```
"""
Ordered-probit hybrid mode choice model: simultaneous maximum likelihood estimation
==========================================================================

This example estimates a hybrid mode choice model with one latent variable and
ordered-probit measurement equations. The observed Likert indicators are treated
as ordinal responses. Their thresholds, the latent-variable model, and the
choice-model parameters are estimated jointly.

The model is the ordered-probit counterpart of
``plot_h05_mode_lv_ordlogit_simult``. The hybrid structure, the latent variable,
the choice utilities, and the simultaneous maximum-likelihood estimation
strategy are the same. Only the ordinal measurement distribution changes: the
indicator probabilities are based on normal cumulative distributions instead of
logistic cumulative distributions.

The latent-variable structure is imported from a separate semantic specification
file. This script adds the ordered-probit measurement configuration, the
normalization constraints required for identification, the mode-choice utilities,
and the simultaneous maximum-likelihood estimation setup.

The script performs the following steps:

- load the latent-variable and indicator specifications,
- define an ordered-probit measurement configuration for all indicators,
- define the normalization constraints used for identification,
- resolve the semantic specification into an estimable model,
- build the Biogeme expressions from the resolved model,
- build the choice utilities, including the latent-variable term,
```

```
- combine the ordered-probit measurement and choice conditional likelihoods,
- integrate the combined conditional likelihood over the latent variable,
- estimate the hybrid model, or reload previously saved estimation results,
- display the estimated parameters as grouped pandas and LaTeX tables.

Michel Bierlaire
Sat Jun 13 2026, 15:13:40
"""

from __future__ import annotations

from choice_latent_variables import generate_availability, generate_utility_functions
from likert_spec import likert_indicators, likert_types
from number_of_draws import NUMBER_OF_DRAWS
from one_latent_variable_spec import latent_variables
from optima import Choice, read_data

import biogeme.biogeme_logging as blog
from biogeme.biogeme import BIOGEME
from biogeme.expressions import MonteCarlo, log
from biogeme.latent_variables import (
    BuildContext,
    EstimationMode,
    Fixing,
    IndicatorMeasurementSpec,
    MeasurementConfiguration,
    MeasurementIntercept,
    MeasurementLoading,
    MeasurementModel,
    MeasurementSigma,
    NormalizationPlan,
    PositiveParameterSpec,
    PositivityMode,
    build_biogeme_model,
    resolve_model,
)
from biogeme.models import logit
from biogeme.results_processing import (
    get_latex_estimated_parameters,
    get_latex_general_statistics,
    get_pandas_estimated_parameters,
)

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

DEFAULT_MEASUREMENT_SIGMA_START = 10.0


# %%
# Ordered-probit measurement configuration for all Likert indicators.
# ---------------------------------------------------------------
# The semantic specification, the latent variable, and the mode-choice component
# are the same as in the ordered-logit example. Only the ordinal measurement
# distribution changes: ordered logit uses the logistic CDF, whereas ordered
# probit uses the normal CDF.
measurement_configuration = MeasurementConfiguration(
    specifications=[
        IndicatorMeasurementSpec(
            indicator_name=indicator.name,
            measurement_model=MeasurementModel.ORDERED_PROBIT,
            measurement_sigma=PositiveParameterSpec(
                start=DEFAULT_MEASUREMENT_SIGMA_START
            ),
        )
        for indicator in likert_indicators
    ]
)

# %%
# Load the Optima data.
# ---------------------
database = read_data()

# %%
# Define the build context for maximum likelihood estimation.
# -----------------------------------------------------------
# The build context specifies how the semantic latent-variable specification is
# translated into estimable Biogeme expressions. We start from the default
# maximum-likelihood context and explicitly select the log-exp parameterization
# for positive parameters. The ordinal options control the construction and
# numerical treatment of ordered-probit thresholds.
default_context = BuildContext.default(EstimationMode.MAXIMUM_LIKELIHOOD)
context = BuildContext(
    estimation_mode=default_context.estimation_mode,
    draw_type=default_context.draw_type,
    positivity_mode=PositivityMode.LOG_EXP,
    naming=default_context.naming,
    ordinal_eps=default_context.ordinal_eps,
    ordinal_enforce_order=default_context.ordinal_enforce_order,
)

# %%
# Identification constraints for the latent variable
# ---------------------------------------------------
# Two layers of normalization are needed in this ordered-probit specification.
# They are the same as in :mod:`plot_h05_mode_lv_ordlogit_simult`.
#
# 1. Ordinal measurement-scale normalization
#    The latent response underlying the ordinal indicators has an arbitrary
#    scale. We therefore fix one measurement sigma to 1.0 in order to anchor
#    the overall scale of the ordered-probit measurement model.
#
# 2. Latent-variable normalization by reference indicator
#    The latent variable itself is not directly observed, so its location and
#    orientation must also be fixed. We use indicator ``Envir01`` as the
#    reference indicator:
#
```

```
#    - its intercept is fixed to 0.0, which anchors the location of the latent
#      variable;
#    - its loading on ``car_centric_attitude`` is fixed to -1.0, which anchors
#      the scale and fixes the orientation of the latent variable.
#
# The negative sign means that higher values of ``car_centric_attitude`` imply
# lower expected values for ``Envir01``. With these constraints in place, the
# remaining intercepts, loadings, and thresholds are interpreted relative to
# the chosen reference and measurement scale.
normalization_plan = NormalizationPlan()

normalization_plan.add(
    Fixing(
        MeasurementSigma('Envir01'),
        1.0,
        note='ordered-probit measurement model: scale normalization',
    )
)

normalization_plan.add(
    Fixing(
        MeasurementIntercept('Envir01'),
        0.0,
        note='car_centric_attitude reference indicator: location',
    )
)
normalization_plan.add(
    Fixing(
        MeasurementLoading('car_centric_attitude', 'Envir01'),
        -1.0,
        note='car_centric_attitude reference indicator: scale and orientation',
    )
)

# %%
# Resolve the semantic specification.
# -----------------------------------
# The resolver combines the latent-variable specification, the indicator
# definitions, the ordered-probit measurement configuration, and the
# normalization plan into an internal resolved model.
resolved_model = resolve_model(
    latent_variables=latent_variables,
    likert_indicators=likert_indicators,
    likert_types=likert_types,
    measurement_configuration=measurement_configuration,
    context=context,
    normalization_plan=normalization_plan,
)

# %%
# Build the Biogeme expressions.
# ------------------------------
# The builder translates the resolved model into expressions that can be used by
# Biogeme for simultaneous maximum-likelihood estimation. It also provides
# report-ready parameter groups based on the parameters that are actually
# estimated.
built_model = build_biogeme_model(resolved_model)

# %%
# Choice utilities including the latent-variable term.
# ----------------------------------------------------
utilities = generate_utility_functions(built_model.latent_expressions)

# %%
# Conditional likelihood of the mode choice model
# -----------------------------------------------
availability = generate_availability()
conditional_choice_likelihood = logit(utilities, availability, Choice)

# %%
# Combined conditional likelihood.
# --------------------------------
# The ordered-probit measurement component and the mode-choice component are
# multiplied conditionally on the latent variable. The product is later
# integrated over the latent-variable distribution, so all components are
# estimated jointly in one likelihood.
combined_conditional_likelihood = (
    built_model.conditional_likelihood * conditional_choice_likelihood
)

# %%
# Log-likelihood.
# ---------------
# The combined conditional likelihood is integrated over the latent variable by
# Monte Carlo. Taking the logarithm of the resulting integrated likelihood yields
# the log-likelihood used for simultaneous maximum-likelihood estimation.
integrated_likelihood = MonteCarlo(combined_conditional_likelihood)
log_likelihood = log(integrated_likelihood)

# %%
# Estimate the model with Biogeme.
# --------------------------------
# Existing results are reloaded from the YAML file when available.
biogeme = BIOGEME(
    database,
    log_likelihood,
    number_of_draws=NUMBER_OF_DRAWS,
    calculating_second_derivatives='never',
    max_iterations=5_000,
    group_of_parameters=built_model.parameter_groups,
)
biogeme.model_name = 'plot_h06_mode_lv_ordprobit_simult'

yaml_file_name = f'saved_results/{biogeme.model_name}.yaml'
results = biogeme.estimate_or_load(yaml_file_name=yaml_file_name)

# %%
```

```
# Display a compact summary and the estimated parameters.
print(results.short_summary())
pandas_results = get_pandas_estimated_parameters(
    estimation_results=results,
    group_of_parameters=built_model.parameter_groups,
)
for group_name, pandas_table in pandas_results.items():
    print(group_name if group_name else 'Estimated parameters')
    print(pandas_table)

general_statistics = get_latex_general_statistics(estimation_results=results)
print(general_statistics)

estimated_parameters = get_latex_estimated_parameters(
    estimation_results=results,
    group_of_parameters=built_model.parameter_groups,
)
for group_name, latex_table in estimated_parameters.items():
    print(group_name if group_name else 'Estimated parameters')
    print(latex_table)
```

The script plot_h06_mode_lv_ordprobit_simult.py implements the ordered-probit version of the simultaneous hybrid choice model. Its structure is deliberately parallel to plot_h05_mode_lv_ordlogit_simult.py: the latent-variable specification, the normalization constraints, the choice utilities, and the simultaneous estimation strategy are the same. The only conceptual difference is the distribution used in the ordinal measurement model. Here, the probabilities of the Likert responses are computed with the normal cumulative distribution function, leading to an ordered-probit measurement model.

The script first imports the semantic specification of the latent variable, the Likert indicators, the number of draws used for Monte-Carlo integration, and the data preparation routine. It then defines a MeasurementConfiguration in which each Likert indicator is associated with an ORDERED_PROBIT measurement model. Positive measurement scale parameters are represented using a log-exp transformation, which ensures that the corresponding standard deviations remain strictly positive during estimation.

The normalization plan contains the identifying restrictions required by the ordered-probit hybrid model. The measurement scale is anchored by fixing the scale parameter of the reference indicator Envir01 to one. The latent variable is normalized using the same reference indicator: its intercept is fixed to zero and its loading on the car-centric attitude is fixed to -1. These constraints determine the location, scale, and orientation of the latent variable.

The semantic specification is then resolved into an estimable model and translated into Biogeme expressions. The choice utilities are generated from the latent-variable expressions, so that the car-centric attitude can enter the utility of the car alternative. The conditional likelihood of the ordered-probit measurement model is multiplied by the conditional logit choice probability. This product is conditional on the latent variable and is integrated by Monte-Carlo simulation. The logarithm of this integrated likelihood is the objective function used for simultaneous maximum-likelihood estimation.

Finally, the script estimates the model, or reloads previously saved results if available. The estimated parameters are then displayed in grouped pandas and LaTeX tables, using the parameter groups generated from the resolved model. This organization makes explicit the correspondence between the theoretical components of the hybrid choice model—structural equation, measurement equations, thresholds, and choice model—and the blocks of parameters reported in the estimation output.

B.12 plot_h07_mode_lv_ordprobit_simult_generated.py

```
"""Simultaneous hybrid choice model with ordered-probit indicators
=================================================================


This example estimates a hybrid choice model where a latent variable,
interpreted as a car-centric attitude, influences the utilities of a
mode-choice model. The latent variable is described by a structural equation
and is measured through several attitudinal indicators using ordered-probit
measurement equations.

The model is estimated simultaneously: the measurement equations and the
choice model are combined into one likelihood function. Because the latent
variable is not observed, the likelihood is integrated by Monte Carlo
simulation.

The script is organized with ``# %%`` markers. Each marker denotes the start
of a new cell when the example is later converted into a notebook. New cells are
introduced whenever a new concept is presented, so that the generated notebook
```

```
can be read progressively as a pedagogical example. The comments therefore
describe both the statistical role of each block and the pedagogical message of
the corresponding notebook cell.

Tested with Biogeme 3.3.3.
Michel Bierlaire
Sat Jun 13 2026, 15:13:40

"""

from choice_latent_variables import (
    generate_availability,
    generate_utility_functions,
)
from number_of_draws import NUMBER_OF_DRAWS
from optima import Choice, read_data

import biogeme.biogeme_logging as blog
from biogeme.biogeme import BIOGEME
from biogeme.expressions import (
    Beta,
    Draws,
    MonteCarlo,
    MultipleProduct,
    MultipleSum,
    OrderedProbit,
    Variable,
    exp,
    log,
)
from biogeme.models import logit
from biogeme.results_processing import (
    get_latex_estimated_parameters,
    get_latex_general_statistics,
    get_pandas_estimated_parameters,
)

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

# %%
# Load the Optima data set.
# -------------------------
# The data set contains the observed mode choices, explanatory variables, and
# attitudinal indicators used in the hybrid choice model.
database = read_data()


# %%
# Model parameters
# ----------------
# The parameters are organized according to their role in the model: thresholds
# for the Likert-scale indicators, standard deviations of measurement errors,
# coefficients of the measurement equations, and coefficients of the structural
# equation of the latent variable.
likert_delta_0_log = Beta('likert_delta_0_log', -0.86, None, None, 0)
likert_delta_0 = exp(likert_delta_0_log)
likert_delta_1_log = Beta('likert_delta_1_log', -0.43, None, None, 0)
likert_delta_1 = exp(likert_delta_1_log)
measurement_Envir01_sigma = 1.0
measurement_Envir02_sigma_log = Beta(
    'measurement_Envir02_sigma_log', 2.302585092994046, None, None, 0
)
measurement_Envir02_sigma = exp(measurement_Envir02_sigma_log)
measurement_Envir06_sigma_log = Beta(
    'measurement_Envir06_sigma_log', 2.302585092994046, None, None, 0
)
measurement_Envir06_sigma = exp(measurement_Envir06_sigma_log)
measurement_LifSty07_sigma_log = Beta(
    'measurement_LifSty07_sigma_log', 2.302585092994046, None, None, 0
)
measurement_LifSty07_sigma = exp(measurement_LifSty07_sigma_log)
measurement_Mobil03_sigma_log = Beta(
    'measurement_Mobil03_sigma_log', 2.302585092994046, None, None, 0
)
measurement_Mobil03_sigma = exp(measurement_Mobil03_sigma_log)
measurement_Mobil05_sigma_log = Beta(
    'measurement_Mobil05_sigma_log', 2.302585092994046, None, None, 0
)
measurement_Mobil05_sigma = exp(measurement_Mobil05_sigma_log)
measurement_Mobil08_sigma_log = Beta(
    'measurement_Mobil08_sigma_log', 2.302585092994046, None, None, 0
)
measurement_Mobil08_sigma = exp(measurement_Mobil08_sigma_log)
measurement_Mobil09_sigma_log = Beta(
    'measurement_Mobil09_sigma_log', 2.302585092994046, None, None, 0
)
measurement_Mobil09_sigma = exp(measurement_Mobil09_sigma_log)
measurement_Mobil10_sigma_log = Beta(
    'measurement_Mobil10_sigma_log', 2.302585092994046, None, None, 0
)
measurement_Mobil10_sigma = exp(measurement_Mobil10_sigma_log)
measurement_coefficient_car_centric_attitude_Envir01 = -1.0
measurement_coefficient_car_centric_attitude_Envir02 = Beta(
    'measurement_coefficient_car_centric_attitude_Envir02', 0.0, None, None, 0
)
measurement_coefficient_car_centric_attitude_Envir06 = Beta(
    'measurement_coefficient_car_centric_attitude_Envir06', 0.0, None, None, 0
)
measurement_coefficient_car_centric_attitude_LifSty07 = Beta(
    'measurement_coefficient_car_centric_attitude_LifSty07', 0.0, None, None, 0
)
measurement_coefficient_car_centric_attitude_Mobil03 = Beta(
    'measurement_coefficient_car_centric_attitude_Mobil03', 0.0, None, None, 0
)
measurement_coefficient_car_centric_attitude_Mobil05 = Beta(
    'measurement_coefficient_car_centric_attitude_Mobil05', 0.0, None, None, 0
)
```

```python
measurement_coefficient_car_centric_attitude_Mobil08 = Beta(
    'measurement_coefficient_car_centric_attitude_Mobil08', 0.0, None, None, 0
)
measurement_coefficient_car_centric_attitude_Mobil09 = Beta(
    'measurement_coefficient_car_centric_attitude_Mobil09', 0.0, None, None, 0
)
measurement_coefficient_car_centric_attitude_Mobil10 = Beta(
    'measurement_coefficient_car_centric_attitude_Mobil10', 0.0, None, None, 0
)
measurement_intercept_Envir01 = 0.0
measurement_intercept_Envir02 = Beta(
    'measurement_intercept_Envir02', 0.0, None, None, 0
)
measurement_intercept_Envir06 = Beta(
    'measurement_intercept_Envir06', 0.0, None, None, 0
)
measurement_intercept_LifSty07 = Beta(
    'measurement_intercept_LifSty07', 0.0, None, None, 0
)
measurement_intercept_Mobil03 = Beta(
    'measurement_intercept_Mobil03', 0.0, None, None, 0
)
measurement_intercept_Mobil05 = Beta(
    'measurement_intercept_Mobil05', 0.0, None, None, 0
)
measurement_intercept_Mobil08 = Beta(
    'measurement_intercept_Mobil08', 0.0, None, None, 0
)
measurement_intercept_Mobil09 = Beta(
    'measurement_intercept_Mobil09', 0.0, None, None, 0
)
measurement_intercept_Mobil10 = Beta(
    'measurement_intercept_Mobil10', 0.0, None, None, 0
)
struct_car_centric_attitude_car_oriented_parents = Beta(
    'struct_car_centric_attitude_car_oriented_parents', 0.0, None, None, 0
)
struct_car_centric_attitude_high_education = Beta(
    'struct_car_centric_attitude_high_education', 0.0, None, None, 0
)
struct_car_centric_attitude_intercept = Beta(
    'struct_car_centric_attitude_intercept', 0.0, None, None, 0
)
struct_car_centric_attitude_low_education = Beta(
    'struct_car_centric_attitude_low_education', 0.0, None, None, 0
)
struct_car_centric_attitude_sigma_log = Beta(
    'struct_car_centric_attitude_sigma_log', 2.302585092994046, None, None, 0
)
struct_car_centric_attitude_sigma = exp(struct_car_centric_attitude_sigma_log)
struct_car_centric_attitude_top_manager = Beta(
    'struct_car_centric_attitude_top_manager', 0.0, None, None, 0
)
struct_car_centric_attitude_used_to_go_to_school_by_car = Beta(
    'struct_car_centric_attitude_used_to_go_to_school_by_car', 0.0, None, None, 0
)

# %%
# Structural equation for the latent variable
# -------------------------------------------
# The latent variable is represented as a normal random variable. Its mean is a
# linear function of observed socio-economic characteristics, and its standard
# deviation is estimated from the data.
mu_car_centric_attitude = (
    struct_car_centric_attitude_intercept
    + struct_car_centric_attitude_top_manager * Variable('top_manager')
    + struct_car_centric_attitude_car_oriented_parents
    * Variable('car_oriented_parents')
    + struct_car_centric_attitude_high_education * Variable('high_education')
    + struct_car_centric_attitude_low_education * Variable('low_education')
    + struct_car_centric_attitude_used_to_go_to_school_by_car
    * Variable('used_to_go_to_school_by_car')
)
draw_car_centric_attitude = Draws(
    'struct_car_centric_attitude_draws', draw_type='NORMAL_MLHS_ANTI'
)
car_centric_attitude = (
    mu_car_centric_attitude
    + struct_car_centric_attitude_sigma * draw_car_centric_attitude
)

# %%
# Threshold system for the ordered-probit indicators
# --------------------------------------------------
# The attitudinal indicators are coded on a five-point Likert scale. The
# ordered-probit model uses four thresholds to separate the five ordered
# response categories. The logarithmic parameterization of the threshold
# increments guarantees the required ordering.
# Threshold system: Likert scale
likert_tau_1 = -(likert_delta_0 + likert_delta_1)
likert_tau_2 = -likert_delta_0
likert_tau_3 = likert_delta_0
likert_tau_4 = likert_delta_0 + likert_delta_1

# %%
# Measurement equations and ordered-probit likelihood terms
# ---------------------------------------------------------
# Each indicator is linked to the latent variable by a linear measurement
# equation. The probability of the observed ordinal answer is then computed
# with an ordered-probit model.
# Indicator: Envir01
mu_Envir01 = (
    measurement_intercept_Envir01
    + measurement_coefficient_car_centric_attitude_Envir01 * car_centric_attitude
)
y_Envir01 = Variable('Envir01')
term_Envir01 = OrderedProbit(
```

```
    eta=mu_Envir01 / measurement_Envir01_sigma,
    cutpoints=[
        likert_tau_1 / measurement_Envir01_sigma,
        likert_tau_2 / measurement_Envir01_sigma,
        likert_tau_3 / measurement_Envir01_sigma,
        likert_tau_4 / measurement_Envir01_sigma,
    ],
    y=y_Envir01,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

# Indicator: Envir02
mu_Envir02 = (
    measurement_intercept_Envir02
    + measurement_coefficient_car_centric_attitude_Envir02 * car_centric_attitude
)
y_Envir02 = Variable('Envir02')
term_Envir02 = OrderedProbit(
    eta=mu_Envir02 / measurement_Envir02_sigma,
    cutpoints=[
        likert_tau_1 / measurement_Envir02_sigma,
        likert_tau_2 / measurement_Envir02_sigma,
        likert_tau_3 / measurement_Envir02_sigma,
        likert_tau_4 / measurement_Envir02_sigma,
    ],
    y=y_Envir02,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

# Indicator: Envir06
mu_Envir06 = (
    measurement_intercept_Envir06
    + measurement_coefficient_car_centric_attitude_Envir06 * car_centric_attitude
)
y_Envir06 = Variable('Envir06')
term_Envir06 = OrderedProbit(
    eta=mu_Envir06 / measurement_Envir06_sigma,
    cutpoints=[
        likert_tau_1 / measurement_Envir06_sigma,
        likert_tau_2 / measurement_Envir06_sigma,
        likert_tau_3 / measurement_Envir06_sigma,
        likert_tau_4 / measurement_Envir06_sigma,
    ],
    y=y_Envir06,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

# Indicator: Mobil03
mu_Mobil03 = (
    measurement_intercept_Mobil03
    + measurement_coefficient_car_centric_attitude_Mobil03 * car_centric_attitude
)
y_Mobil03 = Variable('Mobil03')
term_Mobil03 = OrderedProbit(
    eta=mu_Mobil03 / measurement_Mobil03_sigma,
    cutpoints=[
        likert_tau_1 / measurement_Mobil03_sigma,
        likert_tau_2 / measurement_Mobil03_sigma,
        likert_tau_3 / measurement_Mobil03_sigma,
        likert_tau_4 / measurement_Mobil03_sigma,
    ],
    y=y_Mobil03,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

# Indicator: Mobil05
mu_Mobil05 = (
    measurement_intercept_Mobil05
    + measurement_coefficient_car_centric_attitude_Mobil05 * car_centric_attitude
)
y_Mobil05 = Variable('Mobil05')
term_Mobil05 = OrderedProbit(
    eta=mu_Mobil05 / measurement_Mobil05_sigma,
    cutpoints=[
        likert_tau_1 / measurement_Mobil05_sigma,
        likert_tau_2 / measurement_Mobil05_sigma,
        likert_tau_3 / measurement_Mobil05_sigma,
        likert_tau_4 / measurement_Mobil05_sigma,
    ],
    y=y_Mobil05,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

# Indicator: Mobil08
mu_Mobil08 = (
    measurement_intercept_Mobil08
    + measurement_coefficient_car_centric_attitude_Mobil08 * car_centric_attitude
)
y_Mobil08 = Variable('Mobil08')
term_Mobil08 = OrderedProbit(
    eta=mu_Mobil08 / measurement_Mobil08_sigma,
    cutpoints=[
        likert_tau_1 / measurement_Mobil08_sigma,
        likert_tau_2 / measurement_Mobil08_sigma,
        likert_tau_3 / measurement_Mobil08_sigma,
        likert_tau_4 / measurement_Mobil08_sigma,
    ],
    y=y_Mobil08,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

```

```
# Indicator: Mobil09
mu_Mobil09 = (
    measurement_intercept_Mobil09
    + measurement_coefficient_car_centric_attitude_Mobil09 * car_centric_attitude
)
y_Mobil09 = Variable('Mobil09')
term_Mobil09 = OrderedProbit(
    eta=mu_Mobil09 / measurement_Mobil09_sigma,
    cutpoints=[
        likert_tau_1 / measurement_Mobil09_sigma,
        likert_tau_2 / measurement_Mobil09_sigma,
        likert_tau_3 / measurement_Mobil09_sigma,
        likert_tau_4 / measurement_Mobil09_sigma,
    ],
    y=y_Mobil09,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

# Indicator: Mobil10
mu_Mobil10 = (
    measurement_intercept_Mobil10
    + measurement_coefficient_car_centric_attitude_Mobil10 * car_centric_attitude
)
y_Mobil10 = Variable('Mobil10')
term_Mobil10 = OrderedProbit(
    eta=mu_Mobil10 / measurement_Mobil10_sigma,
    cutpoints=[
        likert_tau_1 / measurement_Mobil10_sigma,
        likert_tau_2 / measurement_Mobil10_sigma,
        likert_tau_3 / measurement_Mobil10_sigma,
        likert_tau_4 / measurement_Mobil10_sigma,
    ],
    y=y_Mobil10,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

# Indicator: LifSty07
mu_LifSty07 = (
    measurement_intercept_LifSty07
    + measurement_coefficient_car_centric_attitude_LifSty07 * car_centric_attitude
)
y_LifSty07 = Variable('LifSty07')
term_LifSty07 = OrderedProbit(
    eta=mu_LifSty07 / measurement_LifSty07_sigma,
    cutpoints=[
        likert_tau_1 / measurement_LifSty07_sigma,
        likert_tau_2 / measurement_LifSty07_sigma,
        likert_tau_3 / measurement_LifSty07_sigma,
        likert_tau_4 / measurement_LifSty07_sigma,
    ],
    y=y_LifSty07,
    categories=[1, 2, 3, 4, 5],
    neutral_labels=[6, -1],
)

# %%
# Conditional measurement likelihood
# ----------------------------------
# Conditional on a realization of the latent variable, the indicator likelihood
# is the product of the ordered-probit probabilities. The corresponding sum of
# logarithms is also constructed for later combination with the choice model.
conditional_measurement_likelihood = MultipleProduct(
    [
        term_Envir01,
        term_Envir02,
        term_Envir06,
        term_LifSty07,
        term_Mobil03,
        term_Mobil05,
        term_Mobil08,
        term_Mobil09,
        term_Mobil10,
    ]
)
conditional_log_likelihood = MultipleSum(
    [
        log(term)
        for term in [
            term_Envir01,
            term_Envir02,
            term_Envir06,
            term_LifSty07,
            term_Mobil03,
            term_Mobil05,
            term_Mobil08,
            term_Mobil09,
            term_Mobil10,
        ]
    ]
)
integrated_measurement_likelihood = MonteCarlo(conditional_measurement_likelihood)


# %%
# Generate the choice utilities.
# ------------------------------
# The latent variable is passed to the utility specification so that the
# car-centric attitude can directly influence the mode-choice probabilities.
latent_expressions = {'car_centric_attitude': car_centric_attitude}
utilities = generate_utility_functions(latent_expressions)

# %%
# Conditional choice likelihood.
# ------------------------------
# Given a realization of the latent variable, the probability of the observed
```

```python
# mode choice is computed using the logit model.
availability = generate_availability()
conditional_choice_likelihood = logit(utilities, availability, Choice)

# %%
# Joint conditional likelihood.
# ------------------------------
# The measurement component and the choice component are combined conditional
# on the latent variable. This expression is the joint contribution of one
# observation before integrating out the unobserved latent variable.
combined_conditional_likelihood = (
    conditional_measurement_likelihood * conditional_choice_likelihood
)

# %%
# Integrate over the latent variable.
# -----------------------------------
# The latent variable is unobserved, so the joint conditional likelihood must
# be integrated over its distribution. The integral is approximated by Monte
# Carlo simulation, and the logarithm of the simulated probability defines the
# log-likelihood used for estimation.
integrated_likelihood = MonteCarlo(combined_conditional_likelihood)
log_likelihood = log(integrated_likelihood)


# %%
# Estimate the simultaneous model with Biogeme.
# ---------------------------------------------
# The dictionary groups parameters in the output tables according to their
# statistical role. Biogeme then estimates the model, or reloads existing
# results from the YAML file when available.
group_of_parmeters = {
    'Structural equation': [
        'struct_car_centric_attitude_intercept',
        'struct_car_centric_attitude_top_manager',
        'struct_car_centric_attitude_car_oriented_parents',
        'struct_car_centric_attitude_high_education',
        'struct_car_centric_attitude_low_education',
        'struct_car_centric_attitude_used_to_go_to_school_by_car',
        'struct_car_centric_attitude_sigma_log',
    ],
    'Measurement equation: Envir02': [
        'measurement_intercept_Envir02',
        'measurement_coefficient_car_centric_attitude_Envir02',
        'measurement_Envir02_sigma_log',
    ],
    'Measurement equation: Envir06': [
        'measurement_intercept_Envir06',
        'measurement_coefficient_car_centric_attitude_Envir06',
        'measurement_Envir06_sigma_log',
    ],
    'Measurement equation: Mobil03': [
        'measurement_intercept_Mobil03',
        'measurement_coefficient_car_centric_attitude_Mobil03',
        'measurement_Mobil03_sigma_log',
    ],
    'Measurement equation: Mobil05': [
        'measurement_intercept_Mobil05',
        'measurement_coefficient_car_centric_attitude_Mobil05',
        'measurement_Mobil05_sigma_log',
    ],
    'Measurement equation: Mobil08': [
        'measurement_intercept_Mobil08',
        'measurement_coefficient_car_centric_attitude_Mobil08',
        'measurement_Mobil08_sigma_log',
    ],
    'Measurement equation: Mobil09': [
        'measurement_intercept_Mobil09',
        'measurement_coefficient_car_centric_attitude_Mobil09',
        'measurement_Mobil09_sigma_log',
    ],
    'Measurement equation: Mobil10': [
        'measurement_intercept_Mobil10',
        'measurement_coefficient_car_centric_attitude_Mobil10',
        'measurement_Mobil10_sigma_log',
    ],
    'Measurement equation: LifSty07': [
        'measurement_intercept_LifSty07',
        'measurement_coefficient_car_centric_attitude_LifSty07',
        'measurement_LifSty07_sigma_log',
    ],
    'Thresholds': ['likert_delta_0_log', 'likert_delta_1_log'],
}

biogeme = BIOGEME(
    database,
    log_likelihood,
    number_of_draws=NUMBER_OF_DRAWS,
    calculating_second_derivatives='never',
    max_iterations=5_000,
    group_of_parameters=group_of_parmeters,
)
biogeme.model_name = 'plot_h07_mode_lv_ordprobit_simult_generated'

yaml_file_name = f'saved_results/{biogeme.model_name}.yaml'
results = biogeme.estimate_or_load(yaml_file_name=yaml_file_name)

# %%
# Report the estimation results.
# ------------------------------
# The compact summary gives the main diagnostics. The estimated parameters and
# general statistics are also exported as pandas and LaTeX tables, which is
# useful when preparing reports, slides, or scientific documents.
print(results.short_summary())
pandas_results = get_pandas_estimated_parameters(
    estimation_results=results,
    group_of_parameters=group_of_parmeters,
```

```
)
for group_name, pandas_table in pandas_results.items():
    print(group_name if group_name else 'Estimated parameters')
    print(pandas_table)

general_statistics = get_latex_general_statistics(estimation_results=results)
print(general_statistics)

estimated_parameters = get_latex_estimated_parameters(
    estimation_results=results,
    group_of_parameters=group_of_parmeters,
)
for group_name, latex_table in estimated_parameters.items():
    print(group_name if group_name else 'Estimated parameters')
    print(latex_table)
```

This script contains an automatically generated implementation of the simultaneous hybrid choice model with ordered-probit measurement equations. It corresponds to the same model structure as `plot_h06_mode_lv_ordprobit_simult.py`, but the code is written explicitly rather than being assembled from higher-level semantic objects. It is therefore useful for inspecting the complete Biogeme expression generated from the model specification.

The script first loads the Optima data set and declares all parameters of the model. These include the threshold parameters of the Likert scale, the scale parameters of the measurement equations, the intercepts and loadings of the indicators, and the coefficients of the structural equation of the latent variable. Positive quantities, such as standard deviations and threshold increments, are parameterized in logarithmic form and exponentiated in the model expression.

The latent variable is then constructed from its structural equation. Its conditional mean is a linear function of socio-economic variables, and its random component is represented using Monte-Carlo draws. The ordered-probit measurement model is built indicator by indicator. For each Likert item, the script defines the latent response, the observed indicator, and the ordered-probit probability associated with the observed response. The four thresholds are generated from two positive increments, ensuring the required ordering of the response categories.

Conditional on a realization of the latent variable, the measurement likelihood is obtained as the product of the ordered-probit probabilities. The same latent variable is then inserted into the mode choice utilities through the function `generate_utility_functions`. The conditional choice probability is computed using the logit model. The measurement and choice components are multiplied to form the joint conditional likelihood, which is then integrated over the latent variable by Monte-Carlo simulation.

The last part of the script estimates the model with Biogeme. The parameters are grouped according to their statistical role, so that the output separates the structural equation, the measurement equations, the threshold parameters, and the choice-model parameters. The script finally prints the estimation summary and exports the estimated parameters both as pandas tables and as LaTeX tables.

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., McFadden, D., Train, K., Walker, J., Bhat, C., Bierlaire, M., Bolduc, D., Boersch-Supan, A., Brownstone, D., Bunch, D., Daly, A., de Palma, A., Gopinath, D., Karlstrom, A. and Munizaga, M. (2002). Hybrid choice models: Progress and challenges, *Marketing Letters* **13**(3): 163–175. DOI: 10.1023/A:1020254301302.

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.