Bayesian Estimation with Biogeme: User Guide

Michel Bierlaire

December 20, 2025

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

Series on Biogeme

Contents

1 Introduction

Biogeme (`biogeme.epfl.ch`) is designed to estimate the parameters of discrete choice models. Although originally built around maximum likelihood estimation, Biogeme also supports Bayesian estimation based on Markov chain Monte Carlo (MCMC), producing draws from the posterior distribution rather than a single point estimate.

This user guide assumes that the reader is familiar with discrete choice models and has successfully installed Biogeme. It focuses on *how to run Bayesian estimation in Biogeme, how to specify models appropriately, and how to interpret the output.* Methodological background (MCMC, HMC, NUTS) is intentionally kept to a minimum.

1.1 When and why Bayesian estimation in Biogeme

Bayesian estimation is particularly useful for:

- mixture models and random coefficients, where classical maximum likelihood requires Monte–Carlo integration;
- latent variable models and complex hierarchical structures;
- applications where full uncertainty propagation is needed (credible intervals, predictive distributions, Bayesian simulation).

1.2 Maximum likelihood vs Bayesian estimation in Biogeme

Although maximum likelihood (ML) and Bayesian estimation rely on the same underlying behavioral model, they differ in interpretation, numerical treatment, and diagnostics. Table 1 summarizes the practical differences from a Biogeme user's perspective.

Table 1: Comparison of maximum likelihood and Bayesian estimation in Biogeme

	Maximum likelihood (ML)	**Bayesian estimation**
Unknown parameters	Fixed but unknown constants	Random variables with prior distributions
Main output	Point estimate (MLE) and asymptotic covariance	Posterior distribution (draws)
Uncertainty interpretation	Asymptotic (large-sample) approximation	Finite-sample uncertainty (conditional on priors)
Estimation algorithm	Deterministic optimization (BFGS, Newton, etc.)	Stochastic simulation (MCMC via NUTS)
Identification issues	Singular or ill-conditioned Hessian	Wide/correlated posterior directions
Random coefficients	Require Monte–Carlo integration	Treated as latent variables (no explicit integration)
Convergence diagnostics	Gradient norm, Hessian eigenvalues	$\widehat{\mathsf{R}}$, ESS, divergences, trace plots
Model comparison	Likelihood ratio tests, AIC, BIC	WAIC, LOO, posterior predictive fit

Rule of thumb. Bayesian computation does not fix identification problems. If a model is weakly identified under ML, the Bayesian posterior will typically be wide, highly correlated, or slow to explore. Identification diagnostics should therefore be interpreted jointly with convergence diagnostics.

2 Quickstart

This section provides the minimal steps to run a Bayesian estimation in BIOGEME and explains key configuration options in `biogeme.toml`. We assume that the model is already specified in Python using standard Biogeme expressions.

2.1 Minimal workflow

Bayesian estimation in Biogeme follows the same high-level workflow as maximum likelihood estimation:

1. specify the model (utilities, likelihood or log-likelihood expression, parameters);
2. configure the estimation algorithm in `biogeme.toml`;
3. run the estimation script;
4. interpret the HTML report and inspect the `.nc` file containing posterior (and optional prior) draws.

In the Bayesian case, Biogeme relies on MCMC sampling (NUTS) to approximate the posterior distribution. The estimation therefore produces *draws* from the posterior distribution rather than a single point estimate. Reported summaries (means, medians, HDIs, diagnostics, information criteria) are computed from these draws.

2.2 Bayesian configuration in `biogeme.toml`

Biogeme reads Bayesian settings from the `[Bayesian]` section of `biogeme.toml`. The most important options control: (i) which sampler backend is used (PyMC vs NumPyro/JAX), (ii) the amount of MCMC computation (warm-up, draws, chains), and (iii) which additional diagnostics and criteria are computed.

Sampler backend and sampling strategy. The option `mcmc_sampling_strategy` controls how sampling is performed:

- `"automatic"`: Biogeme selects a suitable strategy depending on available hardware and installed libraries.
- `"pymc"`: use the standard PyMC NUTS sampler on CPU.
- `"numpyro-parallel"`: use NumPyro/JAX and run one chain per device (multiple CPU devices, GPUs, or TPUs).
- `"numpyro-vectorized"`: use NumPyro/JAX and run all chains vectorized on a single device.

When JAX is available, NumPyro-based strategies can be significantly faster, especially on accelerators or when multiple devices can be used.

Number of chains, warm-up, and posterior draws.

`chains`. Number of independent MCMC chains. A common default is 4. Multiple chains are essential for convergence diagnostics (e.g. $\widehat{\mathsf{R}}$).

`warmup`. Number of warm-up iterations per chain. These iterations are used to adapt the sampler (step size, mass matrix) and are not used for posterior summaries. For difficult posteriors, increasing warm-up is often more effective than increasing the number of retained draws.

`bayesian_draws`. Number of post-warm-up draws *per chain* retained for inference. Increasing this number reduces Monte Carlo error (once the chains mix well), but does not fix non-convergence.

Target acceptance rate. The option `target_accept` is the target acceptance probability for NUTS. Typical values are 0.8–0.9; values such as 0.9 or 0.95 often improve robustness for challenging posteriors (at the cost of smaller step sizes and longer run times). If divergences occur, increasing `target_accept` is a common first adjustment.

Saving prior draws (recommended for identification diagnostics). If `sample_from_prior = "True"`, Biogeme generates prior draws and saves them alongside posterior draws. This is useful to diagnose weak identification, because it enables direct comparisons between prior and posterior dispersion.

Likelihood-based summaries and model comparison criteria.

`calculate_likelihood`. If `"True"`, Biogeme computes likelihood-based statistics derived from posterior draws.

`calculate_waic`. If `"True"`, Biogeme computes WAIC (see Section 5.3).

`calculate_loo`. If `"True"`, Biogeme computes LOO (Pareto-smoothed leave-one-out, see Section 5.3).

2.3 Recommended starting configuration

A reasonable baseline configuration for many discrete choice models is:

- `chains = 4`,
- `warmup = 1000` to `2000`,
- `bayesian_draws = 1000` to `2000`,
- `target_accept = 0.9`,
- `sample_from_prior = "True"` during model development.

2.4 Example `[Bayesian]` section

```
[Bayesian]
mcmc_sampling_strategy = "automatic"
sample_from_prior = "True"
bayesian_draws = 2000
warmup = 2000
chains = 4
target_accept = 0.9
calculate_waic = "True"
calculate_loo = "True"
calculate_likelihood = "True"
```

Practical advice. If you are mainly interested in parameter inference, you may disable WAIC/LOO initially to reduce computation and storage. When comparing models, enable WAIC and/or LOO and ensure that pointwise log-likelihood values are available in the output.

3 Model specification

Model specification for Bayesian estimation is very similar to maximum likelihood specification, with important differences regarding priors and (for mixtures) the treatment of random coefficients.

3.1 Prior distributions

In Biogeme, all unknown model parameters that may be estimated from data are represented by objects of class `Beta`. In the Bayesian context, a `Beta` object plays two simultaneous roles: (i) it defines an unknown quantity appearing in the model expressions (utilities, log-likelihood, etc.), and (ii) it defines how this quantity is treated by the Bayesian sampler through a prior distribution and (optionally) bounds.

The `Beta` constructor. A parameter is created as

```
b_cost = Beta(
    name='b_cost',
    value=-1.0,
    lowerbound=None,
    upperbound=None,
    status=0,
    sigma_prior=10.0,
    prior=None,
)
```

The arguments have the following meaning in Bayesian estimation:

`name` Identifier of the parameter, used in outputs and as the underlying PyMC variable name.

`value` Default value. In Bayesian estimation, this value is used as an initial value for the sampler (via PyMC's `initval`) and as the center of the default prior when no user-defined prior is supplied.

`lowerbound, upperbound` Optional bounds on the support of the parameter. In Bayesian estimation, these bounds matter *twice*: they restrict the admissible parameter space and they also determine whether Biogeme uses a truncated default prior.

`status` If different from 0, the parameter is fixed to its default value `value` and is not sampled. If `status=0`, the parameter is unknown and is estimated/sampled.

`sigma_prior` Scale used by Biogeme for the *default* prior when `prior=None`. It controls how informative the default Normal (or truncated Normal) prior is. Larger values yield weaker regularization.

`prior` User-supplied prior distribution, provided as a *prior factory* (a Python callable). If `prior=None`, Biogeme builds a default prior using `value`, `sigma_prior` and the bounds.

Default priors constructed by Biogeme. If no user-defined prior is provided (`prior=None`), Biogeme constructs:

- an unbounded Normal prior centered at `value` when both bounds are `None`;
- a truncated Normal prior whenever at least one bound is specified.

The prior scale is governed by `sigma_prior`. This default behavior is typically appropriate for most applications and provides mild regularization and numerical stability.

Custom priors via a prior factory. A custom prior is specified by providing a callable that receives: (i) the PyMC variable name, (ii) the initial value, (iii) the lower bound (if any), (iv) the upper bound (if any), and returns a valid PyMC distribution object.

The example below defines a prior factory that constructs a Student-t distribution and truncates it so that the parameter can only take negative values:

```
import pymc as pm
from pytensor.tensor import TensorVariable

def negative_student_prior(
    name: str,
    initial_value: float,
    lower_bound: float | None,
    upper_bound: float | None,
) -> TensorVariable:
    base = pm.StudentT.dist(mu=0.0, sigma=10.0, nu=5.0)
    # Enforce negativity through an upper bound at 0.
    upper = 0.0 if upper_bound is None else min(0.0, upper_bound)
    return pm.Truncated(
        name=name,
        dist=base,
        upper=upper,
        initval=initial_value,
    )
```

The prior is then passed to a Biogeme parameter as follows:

```
b_cost = Beta(
    'b_cost',
    value=-1.0,
    lowerbound=None,
    upperbound=None,
    status=0,
    prior=negative_student_prior,
)
```

Practical recommendations.

- **Avoid uniform priors with NUTS.** Flat densities provide little curvature and can lead to poor exploration, divergences, and slow or unstable convergence.
- **Use bounds to encode identification constraints.** In particular, scale parameters (standard deviations) should be constrained to be positive (e.g. `lowerbound=1e-6`) to avoid artificial sign symmetry and spurious bimodal posteriors.
- **Tune informativeness via `sigma_prior`.** When using the default priors, `sigma_prior` controls the degree of regularization.

3.2 Random coefficients and mixtures

A major practical advantage of Bayesian estimation in Biogeme is that mixture models do not require explicit numerical integration over random parameters. Random coefficients are treated as latent variables and are sampled jointly with structural parameters.

Because integration is handled implicitly by the sampler, the analyst specifies only the *conditional* log-likelihood given a realization of the random coefficients (typically the log of a logit probability). There is no need to write a mixture of logit likelihoods or to manually integrate over the mixing distribution.

In many applications, it is useful to retain simulated draws of random coefficients in the estimation output (heterogeneity analysis, post-estimation simulation, etc.). The following statement tells Biogeme to store the draws of a random coefficient:

```
b_time_rnd = DistributedParameter('b_time_rnd', b_time + b_time_s * b_time_eps)
```

Here, b_time_rnd is the individual-specific realization of the time coefficient, stored for later inspection.

3.3 Scale parameters and identification constraints

In mixture models, scale parameters (e.g. standard deviations of normally distributed random coefficients) must be constrained to avoid artificial sign ambiguities. Without a positivity constraint, the sampler may explore both positive and negative regions that correspond to the same model, producing a spurious bimodal posterior and poor mixing.

In Biogeme, enforce positivity by specifying a small positive lower bound, for example `lowerbound = 1e-6`, when defining the corresponding `Beta` parameter.

3.4 Mixtures with panel data

With panel data, Biogeme assumes that random parameters are drawn *per individual*, not per observation, so all observations belonging to the same person share the same realization of the random coefficients.

From the user's perspective, the model is specified at the observation level. Conditional on random parameters, the contribution of a single observation is simply the logit kernel:

```
log_probability_one_observation = loglogit(v, av, CHOICE)
```

Biogeme automatically aggregates contributions over the observations belonging to the same panel and handles sampling of random coefficients internally.

4 Running Bayesian estimation

In practice, running Bayesian estimation consists of setting the `[Bayesian]` configuration in `biogeme.toml` (Section 2.2) and executing the same estimation script as for ML. Biogeme produces: (i) an HTML report with posterior summaries and diagnostics, and (ii) an `.nc` file containing posterior draws (and optional prior draws).

4.1 Troubleshooting common issues

Chains do not converge ($\widehat{\mathsf{R}} > 1.01$).

- Increase `warmup`.
- Check identification diagnostics (near-linear dependencies, redundancy).
- Consider reparameterization (remove redundant constants, rescale variables).

Low effective sample size (ESS).

- Increase `bayesian_draws`.
- Increase `target_accept`.
- Inspect posterior correlations and consider reparameterization.

Divergent transitions reported.

- Increase `target_accept`.
- Ensure positivity constraints for scale parameters.
- Check for extreme posterior correlations or heavy-tailed priors.

Very slow sampling or excessive run time.

- Reduce complexity during model development.
- Disable WAIC/LOO temporarily.
- Use NumPyro/JAX if available.

Posterior resembles the prior.

- Inspect prior/posterior dispersion ratios.
- Reconsider specification or identifying variation in the data.

5 Interpreting the output

Biogeme performs Bayesian estimation by relying internally on PyMC (and optionally NumPyro/JAX). The output mirrors standard PyMC diagnostics and is reported in a unified HTML report. This section summarizes how to interpret the main outputs. We refer the reader to the online documentation of PyMC and ArviZ, as well as Vehtari et al. (2016), Watanabe (2010) for more information.

5.1 General information about the estimation

Sample size. Number of observations used in the estimation.

Sampler. Biogeme selects an appropriate sampling method depending on hardware and installed libraries. Users can also force a sampling strategy via `mcmc_sampling_strategy`.

Number of chains. Number of independent MCMC chains (typically 4).

Number of draws per chain. Retained post-warm-up draws per chain.

Acceptance rate target. Target acceptance probability for NUTS.

Run time. Total wall-clock time used to obtain the posterior sample.

5.2 Posterior log-likelihood and predictive fit

Log-likelihood at posterior mean. Log-likelihood evaluated at the posterior mean.

Expected log-likelihood. Average of the log-likelihood across posterior draws.

Best-draw log-likelihood. Highest log-likelihood value encountered among posterior draws.

Practical advice. If the log-likelihood at posterior mean and expected log-likelihood differ substantially, the posterior may be wide, skewed, or the model may be sensitive to specific parameter regions.

5.3 Information criteria for model comparison

Biogeme reports two fully Bayesian criteria for model comparison: the Widely Applicable Information Criterion (WAIC) and Pareto-smoothed Leave-One-Out cross-validation (LOO). Both criteria estimate a model's expected predictive performance on new, unobserved data, while accounting for model complexity.

Widely Applicable Information Criterion (WAIC)

WAIC is a Bayesian generalization of the Akaike Information Criterion (AIC). It is based on the *pointwise* log-likelihood contributions and averages predictive performance over the posterior distribution rather than evaluating it at a single point estimate.

Let $\ell_n(\theta) = \log p(y_n \mid \theta)$ denote the log-likelihood contribution of observation n. Given posterior draws $\theta^{(s)}$, $s = 1, \ldots, S$, define:

$$\text{lppd} = \sum_{n=1}^{N} \log\left(\frac{1}{S}\sum_{s=1}^{S} \exp\big(\ell_n(\theta^{(s)})\big)\right),$$

the *log pointwise predictive density*. The effective number of parameters is estimated as

$$p_{\text{WAIC}} = \sum_{n=1}^{N} \text{Var}_s\Big(\ell_n(\theta^{(s)})\Big),$$

where the variance is taken across posterior draws.

WAIC is then defined as

$$\text{WAIC} = -2\left(\text{lppd} - p_{\text{WAIC}}\right).$$

Lower WAIC values indicate better expected out-of-sample predictive performance. Because WAIC relies on posterior draws, it is valid for a wide class of models, including hierarchical and latent-variable models commonly estimated in Biogeme.

Leave-One-Out Cross-Validation (LOO)

LOO estimates predictive performance by repeatedly leaving out one observation and evaluating how well the model predicts it. Formally, it targets

$$\sum_{n=1}^{N} \log p(y_n \mid \mathcal{D}_{-n}),$$

where $\mathcal{D}_{-n}$ denotes the data set with observation n removed.

Exact LOO would require refitting the model N times, which is computationally infeasible. Biogeme therefore relies on *Pareto-smoothed importance sampling* (PSIS-LOO), which approximates leave-one-out predictive densities using the posterior draws from the full data set.

LOO is reported on the deviance scale:

$$\text{LOO} = -2\sum_{n=1}^{N} \log \widehat{p}(y_n \mid \mathcal{D}_{-n}),$$

where $\widehat{p}$ denotes the PSIS approximation.

LOO also provides diagnostic measures (Pareto k values) that assess the reliability of the importance-sampling approximation. Large k values signal influential observations or model misspecification.

Practical interpretation

- Lower WAIC or LOO values indicate better predictive performance.
- Differences should be interpreted relative to their standard errors. As a rule of thumb, differences smaller than about twice the standard error are not decisive.
- LOO is generally more robust than WAIC, especially for complex models with latent variables or weak identification.

In Biogeme, both WAIC and LOO are computed from pointwise log-likelihood values stored in the estimation output. When available, LOO is typically recommended as the primary criterion for Bayesian model comparison.

5.4 Posterior parameter summaries

For each parameter, Biogeme reports statistics derived from posterior draws:

Name. Identifier of the parameter.

Value (posterior mean). Expected value under the posterior distribution.

Median. Posterior median (robust measure of central tendency).

Mode. Posterior mode (kernel density estimation), useful for skewed or multimodal posteriors.

Std err. Posterior standard deviation.

z-value. Mean divided by standard deviation.

p-value. Two-sided posterior probability that the parameter has the opposite sign from its mean.

HDI. Highest Density Interval bounds (typically 95%).

5.5 Convergence diagnostics

$\widehat{\mathsf{R}}$**.** Values close to 1 indicate good mixing. A common threshold is $\widehat{\mathsf{R}} \leqslant 1.01$.

ESS (bulk). Effective sample size for the main mass of the posterior. Values above 400 support reliable estimation of means and variances.

ESS (tail). Effective sample size for tail behavior. Values above 100 help stabilize extreme quantiles.

5.6 Identification diagnostics

These diagnostics detect non-identification or weak identification. Intuitively, identification problems arise when some linear combinations of parameters can vary substantially without affecting the likelihood, leading to very wide posterior directions. Diagnostics use posterior draws and (when available) prior draws.

Posterior covariance diagnostics.

- *Minimum and maximum eigenvalues.* Eigenvalues measure posterior variance along orthogonal directions. A very *large* eigenvalue corresponds to a very wide (nearly flat) direction, indicating weak or non-identification along a linear combination of parameters. A very small eigenvalue corresponds to a tightly concentrated direction.

- *Condition number.* Ratio of largest to smallest eigenvalue; measures anisotropy of the posterior covariance. Large values indicate near-dependencies among parameters. Values around 10^3 deserve attention; 10^5 or more is a strong warning sign.

- *Effective rank.* Effective dimensionality of posterior variability (between 0 and the number of parameters). Values much smaller than the number of parameters suggest that variability concentrates in a lower-dimensional subspace.

Prior covariance diagnostics. Same diagnostics for the prior. If the prior is well behaved but the posterior becomes ill-conditioned, the issue typically originates from the likelihood/-model specification.

Identified by the prior. When prior draws are available, Biogeme reports per-parameter ratios of posterior standard deviation to prior standard deviation:

- ratios close to 1 suggest weak information in the likelihood (prior dominates);
- ratios well below 1 suggest that the data are informative.

5.7 Simulated quantities (`.nc` output)

The `.nc` file stores posterior draws (and optional prior draws) in a PyMC `InferenceData` structure. The explicit list is provided. Key groups include:

constant_data. Observed data treated as fixed (indexed by `Dimension.OBS`).

posterior. Posterior draws of parameters and derived quantities (typically `(chain, draw)`; observation-level quantities include `Dimension.OBS`).

prior. Prior draws, if enabled.

log_likelihood. Pointwise log-likelihood contributions used for WAIC/LOO.

sample_stats. Sampler diagnostics (acceptance rate, divergences, step size, tree depth, energy, etc.).

5.8 Graphical diagnostics

Biogeme includes plots such as trace plots, rank plots, energy plots, and autocorrelation plots, generated by ArviZ. Even when numerical diagnostics look good, graphical inspection may help detect multimodality, slow transitions, or other issues. If the plots are not visible, the user should consider using ArviZ directly to regenerate them separately.

6 Simulation and prediction

Once posterior draws are available, Biogeme supports simulation either at a single representative parameter vector (posterior means) or by propagating full posterior uncertainty.

6.1 Simulation using posterior means

Replace each parameter by its posterior mean and run a deterministic simulation:

```
results = biosim.simulate(the_beta_values=betas)
```

This is fast and interpretable, but ignores posterior uncertainty.

6.2 Bayesian simulation using posterior draws

Propagate uncertainty by simulating the model for a subset of posterior draws:

```
bayesian_results = biosim.simulate_bayesian(
    bayesian_estimation_results=estimation_results,
    percentage_of_draws_to_use=3
)
```

The output becomes a distribution of simulated quantities, enabling credible regions and uncertainty-aware decision support.

References

Vehtari, A., Gelman, A. and Gabry, J. (2016). Practical bayesian model evaluation using leave-one-out cross-validation and waic, *Statistics and Computing* **27**(5): 1413–1432. DOI: 10.1007/s11222-016-9696-4.

Watanabe, S. (2010). Asymptotic equivalence of bayes cross validation and widely applicable information criterion in singular learning theory. DOI: https://doi.org/10.48550/arXiv.1004.2316.