

# Package: choicer (via r-universe)

October 19, 2026

**Title** Discrete Choice Models for Economic Applications

**Version** 0.2.1.9000

**Description** Fast estimation of discrete-choice models for applied economics. Frequentist likelihoods, analytical gradients, and Hessians are implemented in C++ with 'OpenMP' parallelism, scaling efficiently to specifications with many alternative-specific constants. Compiled Gibbs samplers provide Bayesian multinomial probit and hierarchical models. Post-estimation routines cover predicted shares, own- and cross-price elasticities, diversion ratios, willingness to pay, and welfare counterfactuals. Supports multinomial logit ('MNL'), mixed logit ('MXL'), nested logit ('NL'), Bayesian multinomial probit ('MNP'), and hierarchical Bayesian multinomial logit and probit ('HMNL', 'HMNP').

**License** LGPL (>= 3)

**URL** <https://github.com/fpcordeiro/choicer>,  
<https://fpcordeiro.github.io/choicer/>

**BugReports** <https://github.com/fpcordeiro/choicer/issues>

**Encoding** UTF-8

**Depends** R (>= 4.1.0)

**Roxygen** list(markdown = TRUE)

**LinkingTo** Rcpp, RcppArmadillo

**Imports** data.table, graphics, nloptr, randtoolbox (>= 1.31.0), Rcpp, stats, utils

**Suggests** testthat (>= 3.1.7), bit64, numDeriv, future.apply, goftest, knitr, rmarkdown

**VignetteBuilder** knitr

**LazyData** true

**Config/Needs/website** pkgdown

**Config/testthat/edition** 3

**Config/roxygen2/version** 8.0.0  
**Config/pak/sysreqs** cmake  
**Repository** <https://fpcordeiro.r-universe.dev>  
**Date/Publication** 2026-10-19 22:38:45 UTC  
**RemoteUrl** <https://github.com/fpcordeiro/choicer>  
**RemoteRef** HEAD  
**RemoteSha** 93e8da3d79fc1ae9bf54fe5fdb4be4383301750a

## Contents

|                                         |    |
|-----------------------------------------|----|
| blp . . . . .                           | 4  |
| blp.choicer_mnl . . . . .               | 5  |
| blp.choicer_mxl . . . . .               | 6  |
| blp.choicer_nl . . . . .                | 7  |
| blp_contraction . . . . .               | 8  |
| coef.choicer_fit . . . . .              | 10 |
| coef.choicer_hb . . . . .               | 10 |
| coef.choicer_mnp . . . . .              | 11 |
| conditional_tastes . . . . .            | 12 |
| consumer_surplus.choicer_hmnl . . . . . | 14 |
| diversion_ratios.choicer_hb . . . . .   | 17 |
| diversion_ratios.choicer_mnl . . . . .  | 18 |
| diversion_ratios.choicer_mxl . . . . .  | 19 |
| diversion_ratios.choicer_nl . . . . .   | 20 |
| elasticities.choicer_hb . . . . .       | 21 |
| elasticities.choicer_mnl . . . . .      | 22 |
| elasticities.choicer_mxl . . . . .      | 23 |
| elasticities.choicer_nl . . . . .       | 24 |
| ess . . . . .                           | 25 |
| get_halton_normals . . . . .            | 25 |
| gof . . . . .                           | 26 |
| logLik.choicer_fit . . . . .            | 27 |
| logsum.choicer_hmnl . . . . .           | 28 |
| mc_asymptotics . . . . .                | 30 |
| mcse . . . . .                          | 32 |
| mode_choice . . . . .                   | 33 |
| monte_carlo . . . . .                   | 34 |
| mxl_blp_contraction . . . . .           | 35 |
| new_choicer_sim . . . . .               | 37 |
| nl_blp_contraction . . . . .            | 38 |
| nobs.choicer_fit . . . . .              | 39 |
| nobs.choicer_hb . . . . .               | 40 |
| nobs.choicer_mnp . . . . .              | 40 |
| ppc_shares . . . . .                    | 41 |
| predict.choicer_hb . . . . .            | 42 |
| predict.choicer_mnl . . . . .           | 43 |

|                                     |     |
|-------------------------------------|-----|
| predict.choicer_mxl . . . . .       | 45  |
| predict.choicer_nl . . . . .        | 46  |
| prepare_hmnl_data . . . . .         | 48  |
| prepare_hmnp_data . . . . .         | 50  |
| prepare_mnl_data . . . . .          | 52  |
| prepare_mnp_data . . . . .          | 54  |
| prepare_mxl_data . . . . .          | 56  |
| prepare_nl_data . . . . .           | 58  |
| print.choicer_cs . . . . .          | 60  |
| print.choicer_fit . . . . .         | 61  |
| print.choicer_gof . . . . .         | 62  |
| print.choicer_hb . . . . .          | 62  |
| print.choicer_mnp . . . . .         | 63  |
| print.choicer_tastes . . . . .      | 64  |
| print.choicer_wtp . . . . .         | 64  |
| print.summary.choicer_hb . . . . .  | 65  |
| print.summary.choicer_mnl . . . . . | 65  |
| print.summary.choicer_mnp . . . . . | 66  |
| print.summary.choicer_mxl . . . . . | 67  |
| print.summary.choicer_nl . . . . .  | 67  |
| recovery_table . . . . .            | 68  |
| rhat . . . . .                      | 70  |
| run_hmnlogit . . . . .              | 71  |
| run_hmnprobit . . . . .             | 74  |
| run_mnlogit . . . . .               | 76  |
| run_mnprobit . . . . .              | 79  |
| run_mxlogit . . . . .               | 82  |
| run_nestlogit . . . . .             | 90  |
| sample_by_choice . . . . .          | 93  |
| set_num_threads . . . . .           | 95  |
| simulate_hmnl_data . . . . .        | 96  |
| simulate_hmnp_data . . . . .        | 97  |
| simulate_mnl_data . . . . .         | 99  |
| simulate_mnp_data . . . . .         | 100 |
| simulate_mxl_data . . . . .         | 101 |
| simulate_nl_data . . . . .          | 103 |
| summary.choicer_hb . . . . .        | 104 |
| summary.choicer_mnl . . . . .       | 105 |
| summary.choicer_mnp . . . . .       | 106 |
| summary.choicer_mxl . . . . .       | 106 |
| summary.choicer_nl . . . . .        | 107 |
| thread_info . . . . .               | 108 |
| traceplot . . . . .                 | 109 |
| traceplot.choicer_hb . . . . .      | 109 |
| vcov.choicer_fit . . . . .          | 110 |
| vcov.choicer_hb . . . . .           | 112 |
| vcov.choicer_mnp . . . . .          | 112 |
| wesml_vcov . . . . .                | 113 |

|                          |            |
|--------------------------|------------|
| wesml_weights . . . . .  | 114        |
| wtp.choicer_hb . . . . . | 116        |
| <b>Index</b>             | <b>119</b> |

---

|     |                                |
|-----|--------------------------------|
| blp | <i>BLP contraction mapping</i> |
|-----|--------------------------------|

---

**Description**

Finds the ASC (delta) parameters such that predicted market shares match target shares, using the contraction mapping of Berry, Levinsohn, and Pakes (1995) [doi:10.2307/2171802](https://doi.org/10.2307/2171802).

**Usage**

```
blp(object, target_shares, ...)
```

**Arguments**

- object           A fitted model object.
- target\_shares    Numeric vector of target market shares (length J).
- ...              Additional arguments passed to methods.

**Value**

Converged delta (ASC) vector.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
blp(fit, target_shares = rep(1/J, J))
```

---

|                 |                                                            |
|-----------------|------------------------------------------------------------|
| blp.choicer_mnl | <i>BLP contraction mapping for multinomial logit model</i> |
|-----------------|------------------------------------------------------------|

---

## Description

BLP contraction mapping for multinomial logit model

## Usage

```
## S3 method for class 'choicer_mnl'
blp(
  object,
  target_shares,
  delta_init = NULL,
  tol = 1e-08,
  max_iter = 1000,
  ...
)
```

## Arguments

|                            |                                                                                                                                                                                                                          |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>object</code>        | A <code>choicer_mnl</code> object fitted with <code>keep_data = TRUE</code> .                                                                                                                                            |
| <code>target_shares</code> | Numeric vector of target market shares. Length <code>J_inside</code> when no outside option, or <code>J_inside + 1</code> (with the outside option's share at index 1) when <code>include_outside_option = TRUE</code> . |
| <code>delta_init</code>    | Initial guess for delta (ASC) values. If <code>NULL</code> , uses the estimated ASCs from the fitted model (zeros for a model fitted without them, <code>use_asc = FALSE</code> ).                                       |
| <code>tol</code>           | Convergence tolerance (default <code>1e-8</code> ).                                                                                                                                                                      |
| <code>max_iter</code>      | Maximum iterations (default 1000).                                                                                                                                                                                       |
| <code>...</code>           | Additional arguments (ignored).                                                                                                                                                                                          |

## Value

Converged delta (ASC) vector.

## Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
blp(fit, target_shares = rep(1/J, J))
```

---

|                 |                                                      |
|-----------------|------------------------------------------------------|
| blp.choicer_mxl | <i>BLP contraction mapping for mixed logit model</i> |
|-----------------|------------------------------------------------------|

---

## Description

BLP contraction mapping for mixed logit model

## Usage

```
## S3 method for class 'choicer_mxl'
blp(
  object,
  target_shares,
  delta_init = NULL,
  tol = 1e-08,
  max_iter = 1000,
  keep_draws_bytes = 2^32,
  ...
)
```

## Arguments

|                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object           | A choicer_mxl object fitted with keep_data = TRUE.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| target_shares    | Numeric vector of target market shares. Length J_inside when no outside option, or J_inside + 1 (with the outside option's share at index 1) when include_outside_option = TRUE.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| delta_init       | Initial guess for delta (ASC) values. If NULL, uses the estimated ASCs from the fitted model (zeros for a model fitted without them, use_asc = FALSE).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| tol              | Convergence tolerance (default 1e-8).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| max_iter         | Maximum iterations (default 1000).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| keep_draws_bytes | Most memory, in bytes, that the contraction may use to keep the draws across its iterations; it does not change the returned delta. When the draws are formed on the fly and those of all $N$ choice situations take at most this much ( $8K_wSN$ bytes for $K_w$ random coefficients and $S$ draws), the first evaluation of the shares forms them and the later iterations reuse them, which saves up to about a quarter of each iteration on large designs; above it, every iteration forms them again. The default, $2^{32}$ (4 GiB), covers up to about 1.8 million choice situations at $K_w = 3$ and $S = 100$ ; 0 never keeps the draws, and Inf keeps them whenever they are formed on the fly and can be allocated. The memory is taken per call, on top of the fit's data (parallel workers each take their own). Where memory is committed only when used (Linux, macOS, and containers or batch jobs with memory limits), a budget above the memory actually free can end the R session rather than fall back: lower it, or use 0, when memory is tight. |
| ...              | Additional arguments (ignored).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

## Details

The contraction integrates each choice situation over the same draws as `predict()` (for a cross-sectional fit, the estimation draws; a panel fit's estimation draws are per decision maker), formed on the fly, in parallel, and kept across iterations within `keep_draws_bytes`. With stored draws (`draws = "store"`) they are the points of the cube `get_haltan_normals` would build, formed without it; with more than 128 random coefficients, the cube itself, read at every iteration (and nothing more is kept).

## Value

Converged delta (ASC) vector.

## Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mxlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = "x1", random_var_cols = "w1", S = 50L
)
blp(fit, target_shares = rep(1/J, J))
```

---

blp.choicer\_nl

*BLP contraction mapping for nested logit model*


---

## Description

BLP contraction mapping for nested logit model

## Usage

```
## S3 method for class 'choicer_nl'
blp(
  object,
  target_shares,
  delta_init = NULL,
  damping = 1,
  tol = 1e-08,
  max_iter = 1000,
  ...
)
```

**Arguments**

|                            |                                                                                                                                                                                                                          |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>object</code>        | A <code>choicer_n1</code> object fitted with <code>keep_data = TRUE</code> .                                                                                                                                             |
| <code>target_shares</code> | Numeric vector of target market shares. Length <code>J_inside</code> when no outside option, or <code>J_inside + 1</code> (with the outside option's share at index 1) when <code>include_outside_option = TRUE</code> . |
| <code>delta_init</code>    | Initial guess for delta (ASC) values. If <code>NULL</code> , uses the estimated ASCs from the fitted model (zeros for a model fitted without them, <code>use_asc = FALSE</code> ).                                       |
| <code>damping</code>       | Contraction damping factor in $(0, 1]$ (default 1).                                                                                                                                                                      |
| <code>tol</code>           | Convergence tolerance (default $1e-8$ ).                                                                                                                                                                                 |
| <code>max_iter</code>      | Maximum iterations (default 1000).                                                                                                                                                                                       |
| <code>...</code>           | Additional arguments (ignored).                                                                                                                                                                                          |

**Value**

Converged delta (ASC) vector.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, nest := rep(c(1L, 1L, 2L, 2L), N)]
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_nestlogit(dt, "id", "alt", "choice", c("x1", "x2"), "nest")
blp(fit, target_shares = rep(1/J, J))
```

---

blp\_contraction

---

*BLP95 contraction mapping to find delta given target shares*


---

**Description**

BLP95 contraction mapping to find delta given target shares

**Usage**

```
blp_contraction(
  delta,
  target_shares,
  X,
  beta,
  alt_idx,
  M,
```

```

    weights,
    include_outside_option = FALSE,
    tol = 1e-08,
    max_iter = 1000L
  )

```

## Arguments

|                                     |                                                                                                           |
|-------------------------------------|-----------------------------------------------------------------------------------------------------------|
| <code>delta</code>                  | J x 1 vector with initial guess for deltas (ASCs)                                                         |
| <code>target_shares</code>          | J x 1 vector with target shares for each alternative                                                      |
| <code>X</code>                      | sum(M) x K design matrix with covariates. M[i] x K matrix for individual i                                |
| <code>beta</code>                   | K x 1 vector with model parameters                                                                        |
| <code>alt_idx</code>                | sum(M) x 1 vector with indices of alternatives within each choice set; 1-based indexing                   |
| <code>M</code>                      | N x 1 vector with number of alternatives for each individual                                              |
| <code>weights</code>                | N x 1 vector with weights for each observation                                                            |
| <code>include_outside_option</code> | whether to include outside option normalized to 0 (if so, the outside option is not included in the data) |
| <code>tol</code>                    | convergence tolerance                                                                                     |
| <code>max_iter</code>               | maximum number of iterations                                                                              |

## Value

vector with contraction's delta (ASCs) output

## Examples

```

library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
beta <- coef(fit)[fit$param_map$beta]
delta <- blp_contraction(rep(0, J), rep(1/J, J), fit$data$X,
  beta, fit$data$alt_idx, fit$data$M, fit$data$weights)
delta

```

---

|                  |                                                       |
|------------------|-------------------------------------------------------|
| coef.choicer_fit | <i>Extract coefficients from a choicer_fit object</i> |
|------------------|-------------------------------------------------------|

---

**Description**

Extract coefficients from a choicer\_fit object

**Usage**

```
## S3 method for class 'choicer_fit'
coef(object, ...)
```

**Arguments**

|        |                                 |
|--------|---------------------------------|
| object | A choicer_fit object.           |
| ...    | Additional arguments (ignored). |

**Value**

Named numeric vector of estimated coefficients.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
coef(fit)
```

---

|                 |                                                              |
|-----------------|--------------------------------------------------------------|
| coef.choicer_hb | <i>Extract posterior means from a hierarchical Bayes fit</i> |
|-----------------|--------------------------------------------------------------|

---

**Description**

Extract posterior means from a hierarchical Bayes fit

**Usage**

```
## S3 method for class 'choicer_hb'
coef(object, component = c("beta", "theta", "delta", "xi"), ...)
```

**Arguments**

|           |                                                                                                                                                                           |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object    | A choicer_hmnl or choicer_hmnp object.                                                                                                                                    |
| component | Which block to return: "beta" (population means b, default), "theta" (delta mean function), "delta" (alternative effects), or "xi" (unobserved quality, delta - z'theta). |
| ...       | Additional arguments (ignored).                                                                                                                                           |

**Value**

Named numeric vector of posterior means.

**Examples**

```
sim <- simulate_hmnl_data(N = 50, T = 2, J = 3, seed = 42)
fit <- suppressWarnings(run_hmnllogit(sim$data, "task", "alt", "choice", c("x1", "x2"),
  person_col = "pid",
  mcmc = list(R = 300, burn = 100)))
coef(fit)
coef(fit, component = "delta")
```

---

|                  |                                                       |
|------------------|-------------------------------------------------------|
| coef.choicer_mnp | <i>Extract coefficients from a choicer_mnp object</i> |
|------------------|-------------------------------------------------------|

---

**Description**

Returns the posterior means of the identified coefficients ( $\beta/\sqrt{\sigma_{11}}$ , computed per draw).

**Usage**

```
## S3 method for class 'choicer_mnp'
coef(object, ...)
```

**Arguments**

|        |                                 |
|--------|---------------------------------|
| object | A choicer_mnp object.           |
| ...    | Additional arguments (ignored). |

**Value**

Named numeric vector of posterior means.

## Examples

```
library(data.table)
set.seed(42)
N <- 100; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnprobit(dt, "id", "alt", "choice", c("x1", "x2"),
                  mcmc = list(R = 300, burn = 100))
coef(fit)
```

---

|                    |                                                                     |
|--------------------|---------------------------------------------------------------------|
| conditional_tastes | <i>Conditional (individual-level) tastes from a mixed logit fit</i> |
|--------------------|---------------------------------------------------------------------|

---

## Description

For each decision maker, the mean and standard deviation of the random coefficients conditional on the choices they were observed to make,  $E[\beta_n | y_n, x_n; \hat{\theta}]$  (Revelt and Train 2000; Train 2009, ch. 11): the frequentist counterpart of the individual-level `beta_i` summaries of [run\\_hmnllogit](#).

## Usage

```
conditional_tastes(object, ...)

## Default S3 method:
conditional_tastes(object, ...)

## S3 method for class 'choicer_mxl'
conditional_tastes(object, ...)
```

## Arguments

|                     |                                                                                    |
|---------------------|------------------------------------------------------------------------------------|
| <code>object</code> | A <code>choicer_mxl</code> fit with stored data ( <code>keep_data = TRUE</code> ). |
| <code>...</code>    | Unused.                                                                            |

## Details

**Estimand.** The fit estimates the population distribution of tastes,  $f(\beta | \theta)$ . Among decision makers who face  $x_n$  and make the choices  $y_n$ , tastes follow the conditional distribution

$$h(\beta | y_n, x_n; \theta) = \frac{P(y_n | x_n, \beta) f(\beta | \theta)}{\int P(y_n | x_n, \beta) f(\beta | \theta) d\beta},$$

where  $P(y_n | x_n, \beta) = \prod_t P_{nt}(j_{nt} | \beta)$  is the product of the decision maker's logit probabilities. Its mean and standard deviation are simulated at  $\hat{\theta}$  with the estimation draws, as weighted averages over draws  $\beta_{ns}$  with weights  $\omega_{ns} \propto P(y_n | x_n, \beta_{ns})$ . Coefficients are on the scale they enter utility:

$\mu_k + (L\eta)_k$  for a normal coefficient and  $\exp(\mu_k) + \exp((L\eta)_k)$  for the shifted log-normal, with the  $\mu$  term present only when `rc_mean = TRUE`. If a variable's mean is carried instead by the same variable in `covariate_cols` (with `rc_mean = FALSE`), only its random part is reported. This is the utility scale; the `beta_i` summaries of `run_hmnlogit()` are on the scale of the underlying normal, i.e. the log of the coefficient for log-normal coordinates.

**What it is not.** The conditional mean is not the decision maker's own  $\beta_n$ . It is the average taste among people who would make  $n$ 's choices, so it shrinks toward the population mean, the more so the fewer choice situations  $n$  contributes, and its dispersion across decision makers understates the population dispersion. By the law of iterated expectations the conditional means average, in large samples, to the population mean (`mean_cond` vs `model_mean` in population), and by the law of total variance  $Var(\beta) = Var(E[\beta | y]) + E[Var(\beta | y)]$ : the share of taste variance revealed by choices (revealed) and the share left unresolved (residual) should sum to about one. These are numerical checks, not specification tests: for normal random coefficients with free means, both identities are the likelihood's first-order conditions for  $\mu$  and  $\Sigma$ , so they hold at a converged maximum whatever the true taste distribution. A departure points to too few draws  $S$  or an unconverged  $\hat{\theta}$ ; when a random coefficient's mean is fixed at zero (`rc_mean = FALSE` and no fixed coefficient on the same variable), a `mean_cond` far from zero can also flag that restriction. For log-normal coefficients the variance identity holds on the underlying normal scale, not the utility scale reported here, so expect departures of a few percent in either direction at moderate  $S$ . The `sd` is the dispersion of the conditional distribution of tastes, not a standard error: it ignores the sampling error in  $\hat{\theta}$ .

**Cross-section.** Without `person_col` each choice situation is its own decision maker. A single choice reveals little about tastes, so the conditional means stay close to the population mean (revealed small); fit with `person_col` to learn about tastes from repeated choices.

Choice probabilities are handled in log space, so every unit has a finite conditional distribution however improbable its choices. Only a unit whose utilities overflow at the estimates (non-finite log-likelihood) gets NA, and it is left out of the population summaries.

## Value

An object of class `choicer_tastes`: a list with

`mean`, `sd`  $K_w \times U$  matrices of conditional means and standard deviations; rows are the random coefficients (`random_var_cols`), columns the  $U$  likelihood units, named by decision-maker id (`person_col`) or by choice-situation id.

`unit` "decision maker" or "choice situation".

`n_units`  $U$ .

`S` Simulation draws per unit.

`weights` Unit weights used in population.

`population` A data.frame with one row per random coefficient: `model_mean` and `model_sd`, the moments of  $f(\beta | \hat{\theta})$ ; `mean_cond` and `sd_cond`, the weighted mean and standard deviation of the conditional means across units; `revealed` =  $Var(E[\beta | y])/Var(\beta)$ ; and `residual` =  $E[Var(\beta | y)]/Var(\beta)$ .

## References

Revelt, D. and Train, K. (2000). Customer-specific taste parameters and mixed logit: households' choice of electricity supplier. Working Paper E00-274, Department of Economics, University of California, Berkeley.

Train, K. (2009). *Discrete Choice Methods with Simulation*, 2nd ed., Ch. 11. Cambridge University Press.

## See Also

`run_mxlogit` (person\_col), `simulate_mxl_data` (true\_params\$gamma\_i)

## Examples

```
sim <- simulate_mxl_data(N = 150, T = 5, J = 3, seed = 1)
fit <- run_mxlogit(sim$data, "id", "alt", "choice", c("x1", "x2"),
                  c("w1", "w2"), outside_opt_label = 0L,
                  rc_correlation = TRUE, person_col = "pid", S = 50L)
ct <- conditional_tastes(fit)
ct
# conditional means track the realized tastes, with shrinkage
cor(ct$mean["w1", ], sim$true_params$gamma_i["w1", colnames(ct$mean)])
```

---

consumer\_surplus.choicer\_hmnl

*Expected consumer surplus*

---

## Description

Computes the expected consumer surplus per choice situation (Train 2009, Ch. 3):

$$E[CS_i] = \frac{\logsum_i}{-\alpha},$$

where  $\logsum_i$  is the expected maximum utility (see `logsum`) and  $\alpha$  is the (fixed) price coefficient, so that  $-\alpha$  is the marginal utility of income. The formula assumes *no income effects*: utility is linear in price, and the marginal utility of income is constant across the price changes considered.

## Usage

```
## S3 method for class 'choicer_hmnl'
consumer_surplus(
  object,
  price_var,
  newdata = NULL,
  level = 0.95,
  weights = NULL,
  n_draws = 200L,
```

```
    ...
  )

## S3 method for class 'choicer_hmnp'
consumer_surplus(
  object,
  price_var,
  newdata = NULL,
  level = 0.95,
  weights = NULL,
  ...
)

consumer_surplus(
  object,
  price_var,
  newdata = NULL,
  level = 0.95,
  weights = NULL,
  ...
)

## S3 method for class 'choicer_mnl'
consumer_surplus(
  object,
  price_var,
  newdata = NULL,
  level = 0.95,
  weights = NULL,
  ...
)

## S3 method for class 'choicer_mxl'
consumer_surplus(
  object,
  price_var,
  newdata = NULL,
  level = 0.95,
  weights = NULL,
  ...
)

## S3 method for class 'choicer_nl'
consumer_surplus(
  object,
  price_var,
  newdata = NULL,
  level = 0.95,
```

```

    weights = NULL,
    ...
)

```

### Arguments

|           |                                                                                                                                                                                                                                                                                                      |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object    | A fitted model object (choicer_mnl, choicer_mxl, or choicer_nl).                                                                                                                                                                                                                                     |
| price_var | Name of the price variable. Must be a fixed-coefficient variable (a column of the design matrix X).                                                                                                                                                                                                  |
| newdata   | Optional counterfactual data (data.frame or list), as in <a href="#">logsum</a> and <a href="#">predict()</a> . When NULL (default), the data stored at fit time is used (requires keep_data = TRUE).                                                                                                |
| level     | Confidence level for the normal-approximation interval around the mean CS (MNL only). Default 0.95.                                                                                                                                                                                                  |
| weights   | Optional numeric vector with one weight per choice situation, used for the mean CS (and its SE), as in <a href="#">predict()</a> : for a data.frame newdata, one weight per id in order of first appearance. Defaults to equal weights. Ignored when newdata is NULL (the stored fit weights apply). |
| n_draws   | Number of posterior draws to integrate over (hierarchical Bayes methods).                                                                                                                                                                                                                            |
| ...       | Additional arguments passed to methods.                                                                                                                                                                                                                                                              |

### Details

Consumer surplus *levels* inherit the additive utility normalization (in particular the ASC normalization), so the level is only defined up to a constant; *differences* in CS between scenarios — e.g. `consumer_surplus(fit, "price", newdata = scenario)` minus the baseline — are the economically meaningful quantity.

For MNL fits, a delta-method standard error of the weighted mean CS is reported (weights are the stored fit weights, or the resolved newdata weights). For MXL and NL fits only point estimates are returned (`se_mean_cs = NA`): the delta method for the simulated MXL logsum and the nested logsum is deferred; simulation-based intervals (Krinsky-Robb: resample coefficients from their asymptotic distribution and recompute the mean CS) are a practical alternative.

The price variable must have a *fixed* coefficient. For mixed logit a random price coefficient is rejected (as in [wtp](#)): with a random denominator  $1/(-\alpha)$  generally has no finite moments.

### Value

A `choicer_cs` object: a list with `cs` (per-choice- situation surplus, length N, ordered as in [logsum](#)), `mean_cs` (weighted mean), `se_mean_cs` (delta-method SE; NA for MXL/NL or when the variance-covariance matrix is unavailable), `ci` (confidence interval for the mean), `price_var`, `level`, and `n`.

### Methods (by class)

- `consumer_surplus(choicer_hmnl)`: Posterior consumer surplus for the hierarchical logit: per-task logsum divided by the (positive) marginal utility of income  $-\hat{\gamma}_{price}$ , per posterior draw. With newdata, the return also carries the compensating variation against the estimation

`data(attr("cv"))`), i.e. the posterior of  $(\text{logsum}_{new} - \text{logsum}_{base})/(-\tilde{\gamma}_{price})$  summed over tasks. Requires a fixed-sign price coefficient; the posterior-median ratio discipline of `wtp.choicer_hb()` applies. When `newdata` is supplied, `weights` is an optional non-negative length-`n_tasks` vector used in the aggregate CV; equal task weights are the default. The counterfactual must contain the same choice situations as the estimation data, identified by `(person_col, id_col)` (or `id_col` when `person_col = NULL`); rows and tasks may be re-ordered. Unnamed weights follow the baseline tasks' sorted `(person_col, id_col)` order used by the prepared data.

- `consumer_surplus(choicer_hmnp)`: Not available for the probit (see `logsum.choicer_hmnp()`); roadmapped via simulated Emax.

## References

Train, K. (2009). *Discrete Choice Methods with Simulation*, 2nd ed., Ch. 3. Cambridge University Press.

## See Also

[logsum](#), [wtp](#)

## Examples

```
library(data.table)
sim <- simulate_mnl_data(N = 1000, J = 3, beta = c(0.8, -0.6), seed = 123,
                        outside_option = FALSE, vary_choice_set = FALSE)
fit <- run_mnlogit(sim$data, "id", "alt", "choice", c("x1", "x2"))

# treat x2 as the price variable
cs0 <- consumer_surplus(fit, price_var = "x2")
cs0

# Change in consumer surplus from a price increase on alternative 2:
# levels depend on the ASC normalization, differences do not.
dt_cf <- copy(sim$data)[alt == 2, x2 := x2 + 0.5]
cs1 <- consumer_surplus(fit, price_var = "x2", newdata = dt_cf)
delta_cs <- cs1$mean_cs - cs0$mean_cs
delta_cs # negative: the price increase lowers expected surplus
```

---

diversion\_ratios.choicer\_hb

*Compute aggregate diversion ratios*

---

## Description

Computes a  $J \times J$  matrix of diversion ratios. Entry  $(i, j)$  is the fraction of demand lost by alternative  $j$  that is captured by alternative  $i$  when alternative  $j$  becomes less attractive.

**Usage**

```
## S3 method for class 'choicer_hb'
diversion_ratios(object, elast_var, eps = 0.01, n_draws = 100L, ...)

diversion_ratios(object, ...)
```

**Arguments**

|           |                                                               |
|-----------|---------------------------------------------------------------|
| object    | A fitted model object.                                        |
| elast_var | Structural covariate to perturb (hierarchical Bayes methods). |
| eps       | Relative perturbation size (default 0.01).                    |
| n_draws   | Number of posterior draws to integrate over.                  |
| ...       | Additional arguments passed to methods.                       |

**Value**

A  $J \times J$  diversion ratio matrix with alternative labels.

**Methods (by class)**

- `diversion_ratios(choicer_hb)`: Posterior-mean diversion ratios for hierarchical Bayes fits, from the same perturbation engine as `elasticities.choicer_hb()`:  $DR(j \rightarrow k)$  is the fraction of the share alternative  $j$  loses (when its `elast_var` worsens) that flows to  $k$  — including the outside option. Columns are the perturbed alternative  $j$ , rows the receiving alternative  $k$ ; the diagonal is 0.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
diversion_ratios(fit)
```

---

diversion\_ratios.choicer\_mnl

*Diversion ratios for multinomial logit model*

---

**Description**

Diversion ratios for multinomial logit model

**Usage**

```
## S3 method for class 'choicer_mnl'
diversion_ratios(object, ...)
```

**Arguments**

**object**                    A choicer\_mnl object fitted with keep\_data = TRUE.  
**...**                    Additional arguments (ignored).

**Value**

A J x J diversion ratio matrix with alternative labels.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
diversion_ratios(fit)
```

---

diversion\_ratios.choicer\_mxl

*Diversion ratios for mixed logit model*

---

**Description**

Computes the attribute-based diversion ratio matrix. Entry (k, j) is the fraction of demand lost by alternative j that is captured by alternative k when a marginal change in alternative j's wrt\_var attribute reduces s\_j.

**Usage**

```
## S3 method for class 'choicer_mxl'
diversion_ratios(object, wrt_var, is_random_coef = FALSE, ...)
```

**Arguments**

**object**                    A choicer\_mxl object fitted with keep\_data = TRUE.  
**wrt\_var**                   Variable used to perturb alternative j's utility: a column name (character) or 1-based index. Indexes into X columns for fixed coefficients, or W columns for random coefficients (when is\_random\_coef = TRUE).

`is_random_coef` Logical. TRUE if the variable has a random coefficient (is in W), FALSE if fixed (in X). Default FALSE.

... Additional arguments (ignored).

### Details

Unlike MNL, the MXL diversion ratio depends on which variable is perturbed: the realised coefficient  $\beta_{ik}^s$  varies across individuals and draws and does not cancel in the ratio. For a variable with a fixed coefficient the result is independent of the variable ( $\beta$  cancels); for a random-coefficient variable it is not.

### Value

A  $J \times J$  diversion ratio matrix with alternative labels. Cross-products are averaged across simulation draws inside the integration to avoid Jensen-style bias.

### Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mxlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = "x1", random_var_cols = "w1", S = 50L
)
diversion_ratios(fit, "x1")
diversion_ratios(fit, "w1", is_random_coef = TRUE)
```

---

diversion\_ratios.choicer\_nl

*Diversion ratios for nested logit model*

---

### Description

Diversion ratios for nested logit model

### Usage

```
## S3 method for class 'choicer_nl'
diversion_ratios(object, ...)
```

**Arguments**

object            A choicer\_n1 object fitted with keep\_data = TRUE.  
 ...              Additional arguments (ignored).

**Value**

A J x J diversion ratio matrix with alternative labels.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, nest := rep(c(1L, 1L, 2L, 2L), N)]
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_nestlogit(dt, "id", "alt", "choice", c("x1", "x2"), "nest")
diversion_ratios(fit)
```

---

elasticities.choicer\_hb

*Compute aggregate elasticities*

---

**Description**

Computes a J x J matrix of aggregate elasticities. Entry (i, j) is the percentage change in the probability of choosing alternative i when the attribute of alternative j changes by 1\

**Usage**

```
## S3 method for class 'choicer_hb'
elasticities(object, elast_var, eps = 0.01, n_draws = 100L, ...)

elasticities(object, ...)
```

**Arguments**

object            A fitted model object.  
 elast\_var        Structural covariate to perturb (hierarchical Bayes methods).  
 eps              Relative perturbation size (default 0.01).  
 n\_draws          Number of posterior draws to integrate over.  
 ...              Additional arguments passed to methods.

**Value**

A J x J elasticity matrix with alternative labels.

**Methods (by class)**

- `elasticities(choicer_hb)`: Posterior-mean aggregate arc elasticities for hierarchical Bayes fits: each inside alternative's `elast_var` is perturbed by `eps` (default 1%) and shares are re-computed per posterior draw with a common random-coefficient path, giving  $E_{jk} = (\Delta s_j / s_j) / \epsilon$ . Rows are responding alternatives (including the outside option), columns the perturbed alternative.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
```

```
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
elasticities(fit, "x1")
```

---

```
elasticities.choicer_mnl
```

*Elasticities for multinomial logit model*

---

**Description**

Elasticities for multinomial logit model

**Usage**

```
## S3 method for class 'choicer_mnl'
elasticities(object, elast_var, ...)
```

**Arguments**

|                        |                                                                                                           |
|------------------------|-----------------------------------------------------------------------------------------------------------|
| <code>object</code>    | A <code>choicer_mnl</code> object fitted with <code>keep_data = TRUE</code> .                             |
| <code>elast_var</code> | Variable for elasticity computation: a column name (character) or 1-based index into the design matrix X. |
| <code>...</code>       | Additional arguments (ignored).                                                                           |

**Value**

A J x J elasticity matrix with alternative labels.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
elasticities(fit, "x1")
```

---

elasticities.choicer\_mxl

*Elasticities for mixed logit model*

---

**Description**

Elasticities for mixed logit model

**Usage**

```
## S3 method for class 'choicer_mxl'
elasticities(object, elast_var, is_random_coef = FALSE, ...)
```

**Arguments**

|                |                                                                                                                                                                                                    |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object         | A choicer_mxl object fitted with keep_data = TRUE.                                                                                                                                                 |
| elast_var      | Variable for elasticity computation: a column name (character) or 1-based index. Indexes into X columns for fixed coefficients, or W columns for random coefficients (when is_random_coef = TRUE). |
| is_random_coef | Logical. TRUE if the variable has a random coefficient (is in W), FALSE if fixed (in X). Default FALSE.                                                                                            |
| ...            | Additional arguments (ignored).                                                                                                                                                                    |

**Value**

A J x J elasticity matrix with alternative labels.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N))]
dt[, choice := 0L]
```

```
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mxlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = "x1", random_var_cols = "w1", S = 50L
)
elasticities(fit, "x1")
elasticities(fit, "w1", is_random_coef = TRUE)
```

---

elasticities.choicer\_nl

*Elasticities for nested logit model*

---

## Description

Elasticities for nested logit model

## Usage

```
## S3 method for class 'choicer_nl'
elasticities(object, elast_var, ...)
```

## Arguments

|           |                                                                                                           |
|-----------|-----------------------------------------------------------------------------------------------------------|
| object    | A choicer_nl object fitted with keep_data = TRUE.                                                         |
| elast_var | Variable for elasticity computation: a column name (character) or 1-based index into the design matrix X. |
| ...       | Additional arguments (ignored).                                                                           |

## Value

A J x J elasticity matrix with alternative labels.

## Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, nest := rep(c(1L, 1L, 2L, 2L), N)]
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_nestlogit(dt, "id", "alt", "choice", c("x1", "x2"), "nest")
elasticities(fit, "x1")
```

ess

*Rank-normalized effective sample size (bulk and tail)***Description**

Computes the rank-normalized bulk and tail effective sample size (ESS) of Vehtari, Gelman, Simpson, Carpenter & Buerkner (2021, *Bayesian Analysis*) for each column of a matrix of posterior draws. `ess_bulk` measures the effective number of independent draws available for estimating the posterior mean/median; `ess_tail` is the minimum of the 5th- and 95th-percentile indicator ESS, relevant for tail-quantile / credible-interval precision. Autocovariance is computed via `fft` (Geyer's initial monotone sequence estimator of the integrated autocorrelation time).

**Usage**

```
ess(draws)
```

**Arguments**

`draws` A matrix of posterior draws (rows = iterations, columns = parameters) for a single chain, or a list of such matrices (one per chain, identical dimensions).

**Value**

A numeric matrix, one row per parameter, two columns ("bulk", "tail"); NA for parameters with zero variance or a non-finite result.

**Examples**

```
set.seed(42)
draws <- matrix(rnorm(2000), ncol = 2,
                dimnames = list(NULL, c("a", "b")))
ess(draws)
```

get\_halton\_normals

*Halton draws for mixed logit***Description**

Create halton normal draws in appropriate format for mixed logit estimation

**Usage**

```
get_halton_normals(S, N, K_w)
```

**Arguments**

|     |                                                                                                                       |
|-----|-----------------------------------------------------------------------------------------------------------------------|
| S   | Number of draws per draw unit                                                                                         |
| N   | number of draw units: choice situations, or decision makers for a panel fit (person_col in <code>run_mxlogit</code> ) |
| K_w | dimension of random coefficients (number of columns in W matrix)                                                      |

**Details**

Draw unit  $i$  receives the  $S$  consecutive points  $(i - 1)S + 1, \dots, iS$  of the  $K_w$ -dimensional Halton sequence of `randtoolbox::halton()`, mapped to standard normals. The cube takes  $8K_wSN$  bytes and is filled a block of units at a time, so it never holds a second copy of the sequence. Store mode supports at most  $2^{31} - 1$  points ( $S \times N$ ), the largest starting index `halton(start = )` accepts; `draws = "generate"` in `run_mxlogit` has no such limit and never materializes the cube. `predict()`, `logsum()`, `consumer_surplus()`, `elasticities()`, `diversion_ratios()`, `blp()` and `gof()` on a store-mode fit with up to 128 random coefficients do not build their cube (one block per choice situation) with this function: they form the same draws on the fly, from the same uniforms mapped by R's `qnorm()` (bit for bit where `randtoolbox` and `choicer` are compiled with the same floating-point contraction, as on CRAN's platforms).

**Value**

$K_w \times S \times N$  array with halton standard normal draws

**Examples**

```
draws <- get_halton_normals(S = 50, N = 10, K_w = 2)
dim(draws) # 2 x 50 x 10
```

---

|     |                                                  |
|-----|--------------------------------------------------|
| gof | <i>Goodness of fit for a fitted choice model</i> |
|-----|--------------------------------------------------|

---

**Description**

Computes McFadden's pseudo R-squared (plain and adjusted) and the in-sample hit rate for a fitted model.

**Usage**

```
gof(object, null = c("equal_shares", "market_shares"), ...)

## S3 method for class 'choicer_fit'
gof(object, null = c("equal_shares", "market_shares"), ...)
```

**Arguments**

|        |                                                                                                             |
|--------|-------------------------------------------------------------------------------------------------------------|
| object | A fitted model object ( <code>choicer_mnl</code> , <code>choicer_mxl</code> , or <code>choicer_nl</code> ). |
| null   | Null model for the pseudo R-squared: "equal_shares" (default) or "market_shares".                           |
| ...    | Additional arguments passed to methods.                                                                     |

## Details

Two null models are available for the pseudo R-squared  $R^2 = 1 - LL/LL_0$  (adjusted:  $R_{adj}^2 = 1 - (LL - K)/LL_0$  with  $K$  the number of estimated parameters):

- "equal\_shares" (default): every alternative in individual  $i$ 's choice set is equally likely, so  $LL_0 = -\sum_i w_i \log(M_i + 1_{outside})$ . This is exact for unbalanced choice sets and arbitrary weights.
- "market\_shares": the maximized log-likelihood of an ASC-only model,  $LL_0 = \sum_j N_j \log(s_j)$  with  $N_j$  the choice counts and  $s_j$  the observed market shares (including the outside option when present). This closed form is valid only for balanced choice sets and uniform weights; otherwise an error suggests refitting an ASC-only model.

The hit rate is the weighted share of individuals whose observed choice has the highest predicted probability. When the model includes an outside option, the outside good competes for the predicted maximum (its probability is  $1 - \sum_j p_{ij}$ ), and an individual predicted to choose the outside good is a hit when they actually did.

Both the null log-likelihood and the hit rate require the stored estimation data; models fitted with `keep_data = FALSE` return NA fields with a message.

## Value

A `choicer_gof` object: a list with `loglik`, `loglik_null`, `null`, `mcfadden_r2`, `mcfadden_r2_adj`, `hit_rate`, `nobs`, and `n_params`.

## Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
gof(fit)
gof(fit, null = "market_shares")
```

---

|                                 |                                                         |
|---------------------------------|---------------------------------------------------------|
| <code>logLik.choicer_fit</code> | <i>Extract log-likelihood from a choicer_fit object</i> |
|---------------------------------|---------------------------------------------------------|

---

## Description

Returns a `logLik` object, which enables `AIC()` and `BIC()` automatically.

## Usage

```
## S3 method for class 'choicer_fit'
logLik(object, ...)
```

**Arguments**

object            A choicer\_fit object.  
 ...              Additional arguments (ignored).

**Value**

A logLik object with df and nobs attributes.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
logLik(fit)
AIC(fit)
BIC(fit)
```

---

|                     |                                                               |
|---------------------|---------------------------------------------------------------|
| logsum.choicer_hmnl | <i>Expected logsum (inclusive value) per choice situation</i> |
|---------------------|---------------------------------------------------------------|

---

**Description**

Computes the expected maximum utility ("logsum" or inclusive value) for each choice situation, up to the additive constant of the extreme-value error:

- MNL:  $\log \sum_j \exp(V_{ij})$ .
- MXL:  $E_\beta[\log \sum_j \exp(V_{ij}(\beta))]$ , simulated by averaging the log-sum-exp *across* the deterministic Halton draws (regenerated from object\$draws\_info). Taking the log-sum-exp of draw-averaged utilities would understate the expectation (Jensen's inequality), so a dedicated kernel ([mxl\\_logsum](#)) is used.
- NL:  $\log \sum_b \exp(\lambda_b I_{ib})$  with the nest inclusive value  $I_{ib} = \log \sum_{j \in b} \exp(V_{ij}/\lambda_b)$  (singleton nests have  $\lambda_b = 1$ ).

When the model includes an outside option, its normalized utility  $V = 0$  contributes an  $\exp(0)$  term to the sum.

**Usage**

```
## S3 method for class 'choicer_hmnl'
logsum(object, newdata = NULL, n_draws = 200L, ...)

## S3 method for class 'choicer_hmnp'
logsum(object, newdata = NULL, ...)

logsum(object, newdata = NULL, ...)

## S3 method for class 'choicer_mnl'
logsum(object, newdata = NULL, ...)

## S3 method for class 'choicer_mxl'
logsum(object, newdata = NULL, ...)

## S3 method for class 'choicer_nl'
logsum(object, newdata = NULL, ...)
```

**Arguments**

|         |                                                                                                                                                                                                                              |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object  | A fitted model object (choicer_mnl, choicer_mxl, or choicer_nl).                                                                                                                                                             |
| newdata | Optional counterfactual data: a data.frame in the fit-time long format or a list with X, alt_idx, M (plus W for MXL), as in predict(). When NULL (default), the data stored at fit time is used (requires keep_data = TRUE). |
| n_draws | Number of posterior draws to integrate over (hierarchical Bayes methods; thinned evenly from the kept draws).                                                                                                                |
| ...     | Additional arguments passed to methods.                                                                                                                                                                                      |

**Details**

Logsum *levels* depend on the ASC normalization (and, more generally, on any additive utility normalization), so only logsum *differences* between scenarios (e.g. via newdata) are meaningful.

**Value**

Numeric vector with one logsum per choice situation. With a data.frame newdata, choice situations are ordered by id (by decision maker first for a panel mixed logit whose person\_col is present in newdata), as in predict().

**Methods (by class)**

- logsum(choicer\_hmnl): Posterior expected logsum for the hierarchical logit: per choice situation,  $\log(1 + \sum_j \exp V_j)$  against the outside-option anchor, averaged over posterior draws with one  $\beta \sim N(b_r, W_r)$  draw each. Returns the per-task posterior mean vector.
- logsum(choicer\_hmnp): The probit expected maximum has no closed form; simulated-Emax surplus for the HMNP is on the roadmap.

**See Also**[consumer\\_surplus](#)**Examples**

```
library(data.table)
sim <- simulate_mnl_data(N = 500, J = 3, beta = c(0.8, -0.6), seed = 1,
                        outside_option = FALSE, vary_choice_set = FALSE)
fit <- run_mnlogit(sim$data, "id", "alt", "choice", c("x1", "x2"))
head(logsum(fit))
```

mc\_asymptotics

*Asymptotic diagnostics for a Monte Carlo study***Description**

Consumes a `choicer_mc` object and returns per-parameter asymptotic diagnostics: Monte Carlo bias (with MC standard error), empirical SD of the estimates, mean of the reported standard errors, SE-to-SD ratio (information-matrix-equality check), Wald coverage at nominal 90 / 95 / 99 percent with Wilson confidence bands, moments of the studentized statistic  $z = (\theta_{\text{hat}} - \theta_0) / \text{se}$ , and four normality tests on  $z$  (Shapiro-Wilk, Anderson-Darling via `gofest::ad.test`, a hand-coded Jarque-Bera statistic, and a one-sample Kolmogorov-Smirnov test against  $N(0, 1)$ ).

**Usage**

```
mc_asymptotics(
  mc,
  level = 0.95,
  se_col = "se",
  conv_threshold = 0.99,
  se_ratio_threshold_floor = 0.1
)
```

**Arguments**

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>mc</code>     | A <code>choicer_mc</code> object returned by <a href="#">monte_carlo()</a> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <code>level</code>  | Confidence level for the Wilson bands on coverage rates. Defaults to 0.95.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <code>se_col</code> | Name of the column in <code>mc\$replications</code> to use as the standard-error source. Defaults to "se" (the Hessian-based SE stored by <a href="#">monte_carlo()</a> ). Callers that augment replications with an alternative SE flavor (e.g., "se_bhhh" for a BHHH/OPG comparison) can pass that column name to recompute every SE-dependent diagnostic (mean_se, se_ratio, mean_se_w, cov90/95/99, z-moments, normality tests, pass flags) against that flavor. Useful for the information-matrix-equality check in Claim 4 of the MXL validation suite. |

`conv_threshold` Numeric in  $[0, 1]$ . Minimum fraction of replications that must converge for the per-parameter `pass_convergence` flag to be TRUE. The flag compares  $R_{\text{used}} / R_{\text{total}}$  (per parameter) against this threshold. Defaults to 0.99.

`se_ratio_threshold_floor` Numeric scalar. Minimum half-width for the `pass_se_ratio` band. The actual band used is  $\max(\text{se\_ratio\_threshold\_floor}, 3 * 1.4 / \sqrt{R_{\text{used}}})$ , where the  $1.4 / \sqrt{R}$  term approximates the large-sample SD of  $\text{mean\_se} / \text{sd\_emp}$ . The floor guarantees the band is never tighter than the historical hard cutoff. Defaults to 0.10.

## Details

Six logical pass / fail flags are attached to every parameter row: `pass_bias` requires  $|\text{bias\_mc\_se}| < 3$ ; `pass_se_ratio` requires  $|\text{se\_ratio} - 1|$  to lie within  $\max(\text{se\_ratio\_threshold\_floor}, 3 * 1.4 / \sqrt{R_{\text{used}}})$  (a noise-aware band that widens at small  $R_{\text{used}}$  and tightens to the floor at large  $R_{\text{used}}$ ); `pass_cov95` requires the nominal 95 percent level to lie in the Wilson band for empirical coverage; `pass_skew` requires  $|\text{skew\_z}| < 0.3$ ; `pass_kurt` requires excess kurtosis of  $z$  in  $[-0.5, 1.0]$ ; `pass_convergence` requires the per-parameter convergence rate ( $R_{\text{used}} / R_{\text{total}}$ ) to meet `conv_threshold`.

Non-converged replications are excluded per parameter (reported in `R_excluded`). Winsorized (5 percent / 95 percent) versions of `bias`, `sd_emp`, and `mean_se` are reported in parallel columns (`bias_w`, `sd_emp_w`, `mean_se_w`) so silent outlier exclusion is transparent to the reader. Two robust SE-to-SD ratios accompany the Hessian-mean-based `se_ratio`: `se_ratio_med` (median SE divided by the empirical SD) and `se_ratio_w` (winsorized mean SE divided by the winsorized empirical SD); both stay near 1 when 1-2 replications produce near-singular Hessians that inflate `mean_se`. The companion `se_med` column reports the median per-replication SE used by `se_ratio_med`. Neither robust ratio drives a `pass_*` flag — they are purely informational.

Winsorized z-moment counterparts (`mean_z_w`, `sd_z_w`, `skew_z_w`, `kurt_excess_z_w`) are reported alongside the raw z-moments and feed an additional `pass_z_w` flag (Winsorized skew within the same band as `pass_skew` AND Winsorized excess kurtosis within the same band as `pass_kurt`). A companion `pass_cov95_w` flag is TRUE when either `pass_cov95` is TRUE OR the per-rep Winsorized z-CI (the empirical 2.5 / 97.5 percentiles of the Winsorized  $z$ ) covers truth-zero. These two flags are designed for boundary scenarios (e.g., near-zero variance components) where a small number of reps with vanishing SE inflate the raw z-moments without indicating an estimator defect.

## Value

An object of class `choicer_mc_asymptotics` — a `data.table` with one row per unique parameter and columns documented above — with `meta` attached as an attribute (`attr(x, "meta")`).

## Examples

```
sim_fun <- function(seed) simulate_mnl_data(N = 1000, J = 3, seed = seed)
fit_fun <- function(sim) run_mnlogit(
  data = sim$data, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = c("x1", "x2"), outside_opt_label = 0L,
  include_outside_option = FALSE, use_asc = TRUE,
  control = list(print_level = 0L)
)
```

```
mc <- monte_carlo(sim_fun, fit_fun, R = 50L, seed = 1L, progress = FALSE)
mc_asymptotics(mc)
```

mcse

*Monte Carlo standard error of posterior summaries***Description**

Monte Carlo standard error (MCSE) of the posterior mean or median, using the rank-normalized bulk/tail ESS from [ess](#). `kind = "mean"`:  $MCSE = SD / \sqrt{ess\_bulk}$ . `kind = "median"`:  $MCSE = \sqrt{\pi/2} * SD / \sqrt{ess\_tail}$  (the closed-form normal asymptotic-efficiency approximation;  $2/\pi$  is the asymptotic relative efficiency of the sample median vs. the mean under normality).

**Usage**

```
mcse(draws, kind = c("mean", "median"))
```

**Arguments**

|                    |                                                                                                                                                             |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>draws</code> | A matrix of posterior draws (rows = iterations, columns = parameters) for a single chain, or a list of such matrices (one per chain, identical dimensions). |
| <code>kind</code>  | "mean" (default) or "median".                                                                                                                               |

**Value**

Named numeric vector, one value per parameter (NA when the underlying ESS or pooled SD is undefined).

**Examples**

```
set.seed(42)
draws <- matrix(rnorm(2000), ncol = 2,
                dimnames = list(NULL, c("a", "b")))
mcse(draws)
mcse(draws, kind = "median")
```

---

|             |                                     |
|-------------|-------------------------------------|
| mode_choice | <i>Intercity travel mode choice</i> |
|-------------|-------------------------------------|

---

## Description

Stated choices of intercity travel mode for 210 travellers, each choosing among the same four modes: air, train, bus and car. This is the classic Greene & Hensher (1997) data set, reshaped into choicer's long layout (one row per traveller-by-alternative). It is a convenient, recognizable example for multinomial and nested logit models and the demand/welfare toolkit (elasticities, diversion ratios, willingness-to-pay, counterfactuals).

## Usage

```
mode_choice
```

## Format

A data frame with 840 rows (210 travellers x 4 modes) and 9 columns:

**id** Integer traveller (choice situation) identifier, 1-210.

**mode** Factor giving the travel mode: "air", "train", "bus" or "car". Use as the alternative column.

**choice** Integer indicator, 1 for the chosen mode and 0 otherwise. Exactly one mode is chosen per traveller.

**wait** Terminal waiting time in minutes (0 for car).

**travel** In-vehicle travel time in minutes.

**vcost** In-vehicle cost component, in currency units.

**gcost** Generalized cost measure, in currency units.

**income** Household income (traveller level, in thousands).

**size** Size of the travelling party (traveller level).

## Details

wait, travel, vcost and gcost vary across modes within a traveller, while income and size are traveller-level attributes that are constant across modes. A standard specification regresses the choice on wait, travel and vcost with alternative-specific constants; vcost then plays the role of price for willingness-to-pay and consumer-surplus calculations.

The sample is *choice-based*: the survey over-sampled the less popular modes (air, train, bus) and under-sampled car, so sample choice shares do not estimate population mode shares. With a full set of alternative-specific constants the slope coefficients remain consistently estimated under this design (Manski and Lerman, 1977), and willingness-to-pay ratios are unaffected; the constants, and any shares, elasticities or surplus levels computed from fitted probabilities, inherit the sampling design. To target population quantities, attach WESML weights with [wesml\\_weights](#) using external population shares; see `vignette("wesml", package = "choicer")`.

## Source

Greene, W. H. and Hensher, D. A. (1997). Reshaped from the TravelMode data distributed with the **AER** package (<https://CRAN.R-project.org/package=AER>). The same data appear in Greene's *Econometric Analysis* and in several other choice-modelling packages.

## References

Manski, C. F. and Lerman, S. R. (1977). The estimation of choice probabilities from choice based samples. *Econometrica*, 45(8), 1977-1988.

## Examples

```
data(mode_choice)
head(mode_choice)
table(mode_choice$mode[mode_choice$choice == 1L])
```

---

|             |                                       |
|-------------|---------------------------------------|
| monte_carlo | <i>Monte Carlo parameter recovery</i> |
|-------------|---------------------------------------|

---

## Description

Replicates a (DGP -> fit) cycle  $R$  times with independent seeds and collects per-parameter estimates, standard errors, bias, and coverage. Returns a `choicer_mc` object; call `summary()` for aggregated statistics (mean estimate, bias, RMSE, coverage rate, convergence rate).

## Usage

```
monte_carlo(
  sim_fun,
  fit_fun,
  R = 100,
  seed = 1L,
  parallel = FALSE,
  progress = TRUE,
  ...
)
```

## Arguments

|                       |                                                                                                                                                 |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>sim_fun</code>  | Function of seed returning a <code>choicer_sim</code> .                                                                                         |
| <code>fit_fun</code>  | Function of a <code>choicer_sim</code> returning a <code>choicer_fit</code> .                                                                   |
| <code>R</code>        | Number of replications.                                                                                                                         |
| <code>seed</code>     | Base integer seed. Replication $r$ uses $\text{seed} + r - 1L$ .                                                                                |
| <code>parallel</code> | Logical; if TRUE and <code>future.apply</code> is available, run replications in parallel using the user's active <code>future::plan()</code> . |
| <code>progress</code> | Logical; print a one-line progress update per iteration in serial mode. Ignored when <code>parallel = TRUE</code> .                             |
| <code>...</code>      | Unused.                                                                                                                                         |

**Details**

Each iteration calls `sim_fun(seed = seed + r - 1L)`, then `fit_fun(sim)`. Write `sim_fun` as a closure that captures `N`, `J`, and other DGP settings and forwards `seed`. Write `fit_fun` as a closure that takes a `choicer_sim` and returns a fitted `choicer_fit` object, wrapping any data-preparation, draws, or optimizer-control setup.

**Value**

A `choicer_mc` object: a list with elements `replications` (a long `data.table` with one row per estimated parameter per replication) and `meta` (run metadata).

**Examples**

```
sim_fun <- function(seed) simulate_mnl_data(N = 1000, J = 4, seed = seed)
fit_fun <- function(sim) run_mnlogit(
  data = sim$data, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = c("x1", "x2"), outside_opt_label = 0L,
  include_outside_option = FALSE, use_asc = TRUE,
  control = list(print_level = 0L)
)
mc <- monte_carlo(sim_fun, fit_fun, R = 5, seed = 1L, progress = FALSE)
summary(mc)
```

---

|                     |                                                |
|---------------------|------------------------------------------------|
| mxl_blp_contraction | <i>BLP contraction mapping for mixed logit</i> |
|---------------------|------------------------------------------------|

---

**Description**

Finds the ASC (delta) parameters such that predicted market shares match target shares, using the contraction mapping of Berry, Levinsohn, and Pakes (1995).

**Usage**

```
mxl_blp_contraction(
  delta,
  target_shares,
  X,
  W,
  beta,
  mu,
  L_params,
  alt_idx,
  M,
  weights,
  eta_draws,
  rc_dist,
  rc_correlation = TRUE,
```

```

rc_mean = FALSE,
include_outside_option = FALSE,
tol = 1e-08,
max_iter = 1000L,
gen_seed = -1L,
gen_scramble = 1L,
gen_S = 0L
)

```

## Arguments

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| delta                  | J-1 or J vector with initial guess for deltas (ASCs)                                                                                                                                                                                                                                                                                                                                                                                           |
| target_shares          | J vector with target market shares                                                                                                                                                                                                                                                                                                                                                                                                             |
| X                      | design matrix for fixed coefficients; $\text{sum}(M\_i) \times K\_x$                                                                                                                                                                                                                                                                                                                                                                           |
| W                      | design matrix for random coefficients; $\text{sum}(M\_i) \times K\_w$ or $J \times K\_w$                                                                                                                                                                                                                                                                                                                                                       |
| beta                   | $K\_x$ vector with fixed coefficients                                                                                                                                                                                                                                                                                                                                                                                                          |
| mu                     | $K\_w$ vector with mean parameters (raw, will be transformed if log-normal)                                                                                                                                                                                                                                                                                                                                                                    |
| L_params               | Cholesky parameters vector                                                                                                                                                                                                                                                                                                                                                                                                                     |
| alt_idx                | $\text{sum}(M)$ x 1 vector with indices of alternatives; 1-based indexing                                                                                                                                                                                                                                                                                                                                                                      |
| M                      | $N$ x 1 vector with number of alternatives for each individual                                                                                                                                                                                                                                                                                                                                                                                 |
| weights                | $N$ x 1 vector with weights for each observation                                                                                                                                                                                                                                                                                                                                                                                               |
| eta_draws              | Array with draws; $K\_w \times S \times N$                                                                                                                                                                                                                                                                                                                                                                                                     |
| rc_dist                | $K\_w$ vector indicating distribution (0=normal, 1=log-normal)                                                                                                                                                                                                                                                                                                                                                                                 |
| rc_correlation         | whether random coefficients are correlated                                                                                                                                                                                                                                                                                                                                                                                                     |
| rc_mean                | whether mu parameters represent means (TRUE) or are zero (FALSE)                                                                                                                                                                                                                                                                                                                                                                               |
| include_outside_option | whether outside option is included                                                                                                                                                                                                                                                                                                                                                                                                             |
| tol                    | convergence tolerance (default 1e-8)                                                                                                                                                                                                                                                                                                                                                                                                           |
| max_iter               | maximum iterations (default 1000)                                                                                                                                                                                                                                                                                                                                                                                                              |
| gen_seed               | Integer. $< 0$ (default) uses the materialized eta_draws cube; $\geq 0$ forms the draws on the fly (see gen_scramble), with this master seed for the digit permutations of gen_scramble = 1.                                                                                                                                                                                                                                                   |
| gen_scramble           | Integer mode of the draws formed on the fly: 0 = identity permutations (plain Halton) through the generator's inverse normal CDF, 1 = seeded position-wise digit permutations, 2 = identity permutations through R's qnorm(), the store-mode draws of <a href="#">get_halton_normals</a> formed without its cube (bit for bit where randtoolbox and choicer are compiled with the same floating-point contraction). Other values are an error. |
| gen_S                  | Integer number of draws per individual, used only when gen_seed $\geq 0$ .                                                                                                                                                                                                                                                                                                                                                                     |

## Details

With `gen_seed  $\geq 0$`  the draws are formed again at every evaluation of the shares. [blp\(\)](#) on a fitted model runs the same contraction but keeps them across iterations within its `keep_draws_bytes` (see [blp.choicer\\_mxl](#)).

**Value**

vector with converged delta (ASC) values

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
d <- prepare_mxl_data(dt, "id", "alt", "choice", "x1", "w1")
eta <- get_halton_normals(50, d$N, ncol(d$W))
fit <- run_mxlogit(input_data = d, eta_draws = eta)
pm <- fit$param_map
delta <- mxl_blp_contraction(rep(0, J), rep(1/J, J), d$X, d$W,
  coef(fit)[pm$beta], rep(0, ncol(d$W)), coef(fit)[pm$sigma],
  d$alt_idx, d$M, d$weights, eta, rc_dist = rep(0L, ncol(d$W)),
  rc_correlation = FALSE, rc_mean = FALSE)
delta
```

---

new\_choicer\_sim

---

*Construct a choicer\_sim object*


---

**Description**

Wraps simulated data, true parameter values, and DGP settings into a classed list. Returned by [simulate\\_mnl\\_data\(\)](#), [simulate\\_mxl\\_data\(\)](#), and [simulate\\_nl\\_data\(\)](#), and consumed by [recovery\\_table\(\)](#).

**Usage**

```
new_choicer_sim(data, true_params, settings, model)
```

**Arguments**

|             |                                                                                                                                                     |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| data        | A <code>data.table</code> of simulated choice observations.                                                                                         |
| true_params | Named list of true DGP parameters (e.g. <code>beta</code> , <code>delta</code> , <code>Sigma</code> , <code>mu</code> , <code>lambdas</code> ).     |
| settings    | Named list of DGP settings (e.g. <code>N</code> , <code>J</code> , <code>K_x</code> ).                                                              |
| model       | Character scalar: <code>"mnl"</code> , <code>"mxl"</code> , <code>"nl"</code> , <code>"mnp"</code> , <code>"hmn1"</code> , or <code>"hmnp"</code> . |

**Value**

A list of class `choicer_sim`.

---

nl\_blp\_contraction      *BLP95 contraction mapping for the Nested Logit model*


---

### Description

Damped iterative fixed point recovering delta given target shares, using the NL probability structure. damping = 1 reproduces the plain BLP update.

### Usage

```
nl_blp_contraction(
  delta,
  target_shares,
  X,
  beta,
  lambda,
  alt_idx,
  nest_idx,
  M,
  weights,
  include_outside_option = FALSE,
  damping = 1,
  tol = 1e-08,
  max_iter = 1000L
)
```

### Arguments

|                        |                                                                        |
|------------------------|------------------------------------------------------------------------|
| delta                  | J x 1 vector with initial guess for deltas (ASCs).                     |
| target_shares          | vector with target shares (outside-option share first when present).   |
| X                      | sum(M) x K design matrix with covariates.                              |
| beta                   | K x 1 vector with fixed coefficients.                                  |
| lambda                 | full nest dissimilarity vector of length n_nests (singletons = 1).     |
| alt_idx                | sum(M) x 1 vector with indices of alternatives; 1-based indexing.      |
| nest_idx               | J x 1 vector with nest indices for each alternative; 1-based indexing. |
| M                      | N x 1 vector with number of alternatives for each individual.          |
| weights                | N x 1 vector with weights for each observation.                        |
| include_outside_option | whether to include outside option normalized to V=0, lambda=1.         |
| damping                | damping factor for the update (default 1.0 = plain BLP).               |
| tol                    | convergence tolerance.                                                 |
| max_iter               | maximum number of iterations.                                          |

**Value**

vector with contraction's delta (ASCs) output.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, nest := ifelse(alt <= 2, "A", "B")]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_nestlogit(dt, "id", "alt", "choice", c("x1", "x2"), "nest")
beta <- coef(fit)[fit$param_map$beta]
lambda <- rep(1, length(unique(fit$data$nest_idx)))
lambda[as.integer(names(which(table(fit$data$nest_idx) > 1)))] <-
  coef(fit)[fit$param_map$lambda]
delta <- nl_blp_contraction(rep(0, J), rep(1/J, J), fit$data$X, beta, lambda,
  fit$data$alt_idx, fit$data$nest_idx, fit$data$M, fit$data$weights)
delta
```

---

|                  |                                                                 |
|------------------|-----------------------------------------------------------------|
| nobs.choicer_fit | <i>Extract number of observations from a choicer_fit object</i> |
|------------------|-----------------------------------------------------------------|

---

**Description**

Extract number of observations from a choicer\_fit object

**Usage**

```
## S3 method for class 'choicer_fit'
nobs(object, ...)
```

**Arguments**

|        |                                 |
|--------|---------------------------------|
| object | A choicer_fit object.           |
| ...    | Additional arguments (ignored). |

**Value**

Integer number of choice situations.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
nobs(fit)
```

---

|                 |                                                                    |
|-----------------|--------------------------------------------------------------------|
| nobs.choicer_hb | <i>Number of choice situations behind a hierarchical Bayes fit</i> |
|-----------------|--------------------------------------------------------------------|

---

**Description**

Number of choice situations behind a hierarchical Bayes fit

**Usage**

```
## S3 method for class 'choicer_hb'
nobs(object, ...)
```

**Arguments**

|        |                                        |
|--------|----------------------------------------|
| object | A choicer_hmnl or choicer_hmnp object. |
| ...    | Additional arguments (ignored).        |

**Value**

Integer count of choice situations (tasks).

---

|                  |                                                                 |
|------------------|-----------------------------------------------------------------|
| nobs.choicer_mnp | <i>Extract number of observations from a choicer_mnp object</i> |
|------------------|-----------------------------------------------------------------|

---

**Description**

Extract number of observations from a choicer\_mnp object

**Usage**

```
## S3 method for class 'choicer_mnp'
nobs(object, ...)
```

**Arguments**

object            A choicer\_mnp object.  
 ...              Additional arguments (ignored).

**Value**

Integer number of choice situations.

**Examples**

```
library(data.table)
set.seed(42)
N <- 100; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnpnprobit(dt, "id", "alt", "choice", c("x1", "x2"),
                     mcmc = list(R = 300, burn = 100))
nobs(fit)
```

ppc\_shares

*Posterior-predictive share check for hierarchical Bayes fits***Description**

Compares each alternative's observed take rate (share of choice situations in which it was chosen, including the outside option) with its posterior-predictive share from `predict.choicer_hb()`. Large systematic gaps indicate model misfit — e.g. a missing covariate or an outside-option share the delta level cannot rationalize.

**Usage**

```
ppc_shares(object, n_draws = 200L)
```

**Arguments**

object            A choicer\_hmnl or choicer\_hmnp fit (with keep\_data = TRUE).  
 n\_draws          Posterior draws to integrate over (default 200).

**Value**

A data.table with columns alternative, observed, predicted, lower, upper (95% posterior-predictive interval), and covered (is the observed share inside the interval).

## Examples

```
sim <- simulate_hmnl_data(N = 100, T = 3, J = 4, seed = 42)
fit <- suppressWarnings(run_hmnllogit(sim$data, "task", "alt", "choice", c("x1", "x2"),
  person_col = "pid",
  mcmc = list(R = 500, burn = 200)))
ppc_shares(fit)
```

---

|                    |                                                                              |
|--------------------|------------------------------------------------------------------------------|
| predict.choicer_hb | <i>Posterior choice probabilities and shares for hierarchical Bayes fits</i> |
|--------------------|------------------------------------------------------------------------------|

---

## Description

Computes counterfactual choice probabilities, integrating over the posterior draws and (at the population level) over the random-coefficient distribution: for each kept draw  $(b_r, W_r, \delta_r, \dots)$  one  $\beta \sim N(b_r, W_r)$  is drawn and the model probabilities are averaged. Alternatives in newdata that were not in the estimation sample receive a posterior-predictive  $\delta_{new} \sim N(z'_{new}\theta_r, \sigma_{d,r}^2)$  — the entry counterfactual unlocked by the random-effects  $\delta$ . Price or subsidy counterfactuals are just modified covariate columns in newdata.

## Usage

```
## S3 method for class 'choicer_hb'
predict(
  object,
  newdata = NULL,
  level = c("population", "individual"),
  n_draws = 200L,
  aggregate = TRUE,
  ...
)
```

## Arguments

|           |                                                                                                                                                                                                                                             |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object    | A choicer_hmnl or choicer_hmnp fit.                                                                                                                                                                                                         |
| newdata   | Data frame with the estimation columns (choice column not required). NULL (default) predicts on the estimation data.                                                                                                                        |
| level     | "population" (default) integrates over $N(b, W)$ ; "individual" uses the respondent-level $\beta_i$ (requires the prediction rows to belong to estimation respondents; fully posterior-integrated when the fit kept keep_beta_i = "draws"). |
| n_draws   | Number of posterior draws to integrate over (thinned evenly from the kept draws; default 200).                                                                                                                                              |
| aggregate | If TRUE (default) return a per-alternative posterior share table (including the outside option); if FALSE return the posterior-mean probability per row of the prediction data.                                                             |
| ...       | Ignored.                                                                                                                                                                                                                                    |

**Details**

HMNL probabilities are closed-form logit; HMNP probabilities use the 1-D Gauss-Hermite representation of the iid-probit integral  $P(j) = \int \phi(u) \prod_{k \neq j} \Phi(V_j - V_k + u) du$ .

**Value**

With `aggregate = TRUE`, a `data.table` with columns `alternative`, `share` (posterior mean), `sd`, `lower`, `upper` (95% equal-tailed interval); the posterior share draws are attached as `attr(, "draws")`. With `aggregate = FALSE`, a numeric vector of posterior-mean choice probabilities, one per prediction row.

**Examples**

```
sim <- simulate_hmnl_data(N = 100, T = 3, J = 4, seed = 42)
fit <- suppressWarnings(run_hmnlogit(sim$data, "task", "alt", "choice", c("x1", "x2"),
                                   person_col = "pid", alt_covariate_cols = "z1",
                                   mcmc = list(R = 500, burn = 200)))
predict(fit)                                # posterior shares, estimation data
cf <- sim$data
cf$x1 <- cf$x1 + 0.5                        # a counterfactual attribute change
predict(fit, newdata = cf)
```

---

|                     |                                               |
|---------------------|-----------------------------------------------|
| predict.choicer_mnl | <i>Predict from a multinomial logit model</i> |
|---------------------|-----------------------------------------------|

---

**Description**

Computes choice probabilities or aggregate market shares, either for the data used at fit time (default) or for counterfactual newdata.

**Usage**

```
## S3 method for class 'choicer_mnl'
predict(
  object,
  type = c("probabilities", "shares"),
  newdata = NULL,
  weights = NULL,
  ...
)
```

**Arguments**

|                     |                                                                                                       |
|---------------------|-------------------------------------------------------------------------------------------------------|
| <code>object</code> | A <code>choicer_mnl</code> object.                                                                    |
| <code>type</code>   | One of "probabilities" (individual-level choice probabilities) or "shares" (aggregate market shares). |

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>newdata</code> | <p>Optional data for counterfactual prediction. Either:</p> <ul style="list-style-type: none"> <li>• a <code>data.frame</code> in the same long format used at fit time (one row per id-alternative pair, with the fit-time id, alternative, and covariate columns; a choice column is not required). Alternative labels must have been seen at fit time; per-id subsets of alternatives are allowed.</li> <li>• a list with elements <code>X</code>, <code>alt_idx</code>, <code>M</code> (and optionally <code>weights</code>) matching the layout of <code>object\$data</code> — the "modified design matrix" path for policy simulation (e.g., perturb a column of <code>object\$data\$X</code>). <code>alt_idx</code> must use the fit-time integer codes from <code>object\$alt_mapping</code>.</li> </ul> <p>When <code>NULL</code> (default), the data stored at fit time is used (requires <code>keep_data = TRUE</code>).</p> |
| <code>weights</code> | Optional numeric vector with one weight per choice situation, used for <code>type = "shares"</code> aggregation. For a <code>data.frame newdata</code> , supply one weight per id in order of first appearance in <code>newdata</code> (weights are realigned internally to the sorted row order). Defaults to equal weights. Ignored when <code>newdata</code> is <code>NULL</code> (the stored fit weights apply).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <code>...</code>     | Additional arguments (ignored).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## Value

For "probabilities": a list with `choice_prob` and `utility`, one-column matrices with one row per row of the data; with a `data.frame newdata`, rows are ordered by id, then by fit-time alternative code (`alt_int` in `object$alt_mapping`). For "shares": the market shares as a one-column matrix with one row per alternative of `object$alt_mapping`, in its order (the outside option's first when it is modelled), 0 for an alternative that `newdata` does not offer.

## Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
predict(fit, type = "shares")
predict(fit, type = "probabilities")

# Counterfactual: increase x1 for alternative 2
dt_cf <- copy(dt)[alt == 2, x1 := x1 + 1]
predict(fit, type = "shares", newdata = dt_cf)
```

---

predict.choicer\_mxl     *Predict from a mixed logit model*

---

## Description

Computes simulated choice probabilities or aggregate market shares using deterministic Halton draws, either for the data used at fit time (default) or for counterfactual newdata.

## Usage

```
## S3 method for class 'choicer_mxl'
predict(
  object,
  type = c("probabilities", "shares"),
  newdata = NULL,
  weights = NULL,
  ...
)
```

## Arguments

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object  | A choicer_mxl object.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| type    | Either "probabilities" (per-observation simulated choice probabilities) or "shares" (aggregate simulated market shares).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| newdata | Optional data for counterfactual prediction. Either: <ul style="list-style-type: none"> <li>• a data.frame in the same long format used at fit time (one row per id-alternative pair, with the fit-time id, alternative, fixed-coefficient, and random-coefficient columns; a choice column is not required). Alternative labels must have been seen at fit time; per-id subsets of alternatives are allowed.</li> <li>• a list with elements X, W, alt_idx, M (and optionally weights) matching the layout of object\$data — the "modified design matrix" path for policy simulation. alt_idx must use the fit-time integer codes from object\$alt_mapping.</li> </ul> <p>When NULL (default), the data stored at fit time is used (requires keep_data = TRUE). Halton draws are regenerated deterministically from object\$draws_info with one block of draws per choice situation in newdata.</p> |
| weights | Optional numeric vector with one weight per choice situation, used for type = "shares" aggregation. For a data.frame newdata, supply one weight per id in order of first appearance in newdata (weights are realigned internally to the sorted row order). Defaults to equal weights. Ignored when newdata is NULL (the stored fit weights apply).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| ...     | Additional arguments (ignored).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**Value**

For "probabilities": a list with choice\_prob and utility, one-column matrices averaged across simulation draws with one row per row of the data, and with the outside option choice\_prob\_outside, one row per choice situation; with a data.frame newdata, rows are ordered by id, then by fit-time alternative code (alt\_int in object\$alt\_mapping), and for a panel fit whose person\_col is present in newdata, by decision maker first, which reproduces the in-sample row order. For "shares": the simulated market shares as a one-column matrix with one row per alternative of object\$alt\_mapping, in its order (the outside option's first when it is modelled), 0 for an alternative that newdata does not offer. Predictions are unconditional: each choice situation integrates over the population taste distribution (see [conditional\\_tastes\(\)](#) for tastes conditional on observed choices).

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mxlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = "x1", random_var_cols = "w1", S = 50L
)
predict(fit, type = "shares")
predict(fit, type = "probabilities")
```

---

|                    |                                          |
|--------------------|------------------------------------------|
| predict.choicer_nl | <i>Predict from a nested logit model</i> |
|--------------------|------------------------------------------|

---

**Description**

Computes choice probabilities or aggregate market shares, either for the data used at fit time (default) or for counterfactual newdata.

**Usage**

```
## S3 method for class 'choicer_nl'
predict(
  object,
  type = c("probabilities", "shares"),
  newdata = NULL,
  weights = NULL,
  ...
)
```

**Arguments**

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object  | A choicer_nl object.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| type    | One of "probabilities" (individual-level choice probabilities) or "shares" (aggregate market shares).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| newdata | Optional data for counterfactual prediction. Either: <ul style="list-style-type: none"> <li>• a data.frame in the same long format used at fit time (one row per id-alternative pair, with the fit-time id, alternative, and covariate columns; a choice column is not required). Alternative labels must have been seen at fit time; per-id subsets of alternatives are allowed. The alternative-to-nest mapping always comes from the fitted object (it indexes the estimated lambda parameters), so a nest column in newdata is not required and is ignored if present.</li> <li>• a list with elements X, alt_idx, M (and optionally weights) matching the layout of object\$data — the "modified design matrix" path for policy simulation. alt_idx must use the fit-time integer codes from object\$alt_mapping.</li> </ul> When NULL (default), the data stored at fit time is used (requires keep_data = TRUE). |
| weights | Optional numeric vector with one weight per choice situation, used for type = "shares" aggregation. For a data.frame newdata, supply one weight per id in order of first appearance in newdata (weights are realigned internally to the sorted row order). Defaults to equal weights. Ignored when newdata is NULL (the stored fit weights apply).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| ...     | Additional arguments (ignored).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

**Value**

For "probabilities": a list with choice\_prob and utility, one-column matrices with one row per row of the data; with a data.frame newdata, rows are ordered by id, then by fit-time alternative code (alt\_int in object\$alt\_mapping). For "shares": the market shares as a one-column matrix with one row per alternative of object\$alt\_mapping, in its order (the outside option's first when it is modelled), 0 for an alternative that newdata does not offer.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, nest := rep(c(1L, 1L, 2L, 2L), N)]
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_nestlogit(dt, "id", "alt", "choice", c("x1", "x2"), "nest")
predict(fit, type = "shares")
predict(fit, type = "probabilities")
```

---

|                   |                                                                     |
|-------------------|---------------------------------------------------------------------|
| prepare_hmnl_data | <i>Prepare inputs for hierarchical multinomial logit estimation</i> |
|-------------------|---------------------------------------------------------------------|

---

### Description

Prepares and validates panel (or cross-sectional) choice data for the hierarchical Bayesian multinomial logit. The model has two random-effect levels: respondent-level structural tastes  $\beta_i \sim N(b, W)$  over the covariate\_cols, and a global alternative-level effect  $\delta_j = z_j' \theta + \xi_j$ ,  $\xi_j \sim N(0, \sigma_d^2)$ , with mean-function design  $z_j$  built from alt\_covariate\_cols.

### Usage

```
prepare_hmnl_data(
  data,
  id_col,
  alt_col,
  choice_col,
  covariate_cols,
  person_col = NULL,
  alt_covariate_cols = NULL,
  outside_opt_label = NULL,
  cf_residual_col = NULL,
  include_outside_option = TRUE,
  rc_dist = NULL
)
```

### Arguments

|                    |                                                                                                                                                                            |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| data               | Data frame containing choice data.                                                                                                                                         |
| id_col             | Name of the column identifying choice situations (tasks). Task ids only need to be unique within a respondent.                                                             |
| alt_col            | Name of the column identifying alternatives.                                                                                                                               |
| choice_col         | Name of the column indicating the chosen alternative (1 = chosen, 0 = not chosen).                                                                                         |
| covariate_cols     | Vector of names of structural covariate columns (the random-coefficient dimensions).                                                                                       |
| person_col         | Name of the respondent column grouping choice situations. NULL (default) makes each choice situation its own respondent.                                                   |
| alt_covariate_cols | Names of alternative-level covariate columns (constant within each alternative) forming the $\delta$ mean function. NULL (default) gives an intercept-only design (P = 1). |
| outside_opt_label  | Label of physical outside-option rows, removed when include_outside_option = TRUE (the outside good is implicit).                                                          |

|                        |                                                                                                                                                                                                                                                                                                                                   |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| cf_residual_col        | Name of a first-stage residual column (control function for an endogenous covariate), appended to X. Default NULL.                                                                                                                                                                                                                |
| include_outside_option | Logical; if TRUE (default) an implicit outside option with systematic utility 0 is part of every choice set.                                                                                                                                                                                                                      |
| rc_dist                | Integer vector, one entry per column of covariate_cols: 0 for a normal random coefficient, 1 for log-normal (the coefficient enters utility as $\exp(\beta_{ik})$ ; hierarchy normal on the log scale). Default NULL is all-normal. Automatically aligned through dropped columns; a cf_residual_col coordinate is always normal. |

## Details

**Structure.** The design matrix X carries structural covariates only — no alternative-specific-constant dummies. The alternative effect  $\delta_j$  is indexed by `alt_of_row` (integer codes 1 . . J), so memory and compute scale with the number of rows, not with J extra design columns.

**Outside option.** With `include_outside_option = TRUE` (the default) the outside good is modelled implicitly, following the `prepare_mnl_data()` convention: physical outside rows (identified by `outside_opt_label`) are removed, the estimation kernels add the outside term (systematic utility 0), and a choice situation whose inside rows are all 0 in `choice_col` is coded as "outside chosen" (`choice_pos = 0`). The outside option anchors the location of  $\delta$  (mean utility relative to the outside good).

**Cross-section vs panel.** `person_col` groups choice situations into respondents sharing one  $\beta_i$ . With `person_col = NULL` (default) every choice situation is its own respondent (Ti all 1) — the cross-sectional random-coefficients mode.

**Control function.** `cf_residual_col` (a user-supplied first-stage residual, Petrin & Train 2010) is appended to X as an ordinary covariate; its provenance is recorded in `data_spec`. The first stage is NOT run here — supplying a valid residual is the user's responsibility.

## Value

A list of class `c("choicer_data_hmnl", "list")` containing:

- X: Structural design matrix (total\_rows x K\_struct, double), no ASC columns; `cf_residual_col` last when supplied.
- `alt_of_row`: Integer alternative code per row (1 . . J).
- `alt_idx`: Alias of `alt_of_row` for the pooled-MLE init.
- Z: Alternative-level design (J x P), intercept first.
- M: Inside alternatives per choice situation.
- `choice_pos`: 1-based within-task position of the chosen row; 0 = outside option chosen.
- Ti: Choice situations per respondent.
- `person_ids`, `N_persons`, `n_tasks`, `J`, `K_struct`, `P`.
- `include_outside_option`: Logical flag.
- `alt_mapping`: Data.table mapping alternatives to summary statistics (outside option is `alt_int = 0`).

- param\_map: Named list of index vectors (beta, theta), robust to collinearity drops.
- rc\_dist: Integer vector aligned with the columns of X.
- dropped\_cols, dropped\_z\_cols: Dropped column names, if any.
- data\_spec: Column-name metadata (incl. person\_col, outside\_opt\_label, cf\_residual\_col, alt\_covariate\_cols).

### Column names

No column the model uses (the id, alternative, choice, covariate, weight, cluster, nest or decision-maker column) may have a name starting with .choicer\_, which choicer reserves for its working columns, and the alternative column may not be named alt\_int, N\_OBS, N\_CHOICES, TAKE\_RATE or MKT\_SHARE, the names of the other columns of alt\_mapping. Either is an error that names the column. Columns the model does not use may have any name.

### See Also

[prepare\\_hmnp\\_data\(\)](#) for the hierarchical probit counterpart.

### Examples

```
library(data.table)
set.seed(42)
N <- 20; T <- 3; J <- 4
dt <- data.table(
  pid = rep(1:N, each = T * J),
  task = rep(seq_len(N * T), each = J),
  alt = rep(1:J, N * T)
)
dt[, `:=`(x1 = rnorm(.N), x2 = runif(.N, -1, 1))]
dt[, quality := 0.1 * alt] # alternative-level covariate
dt[, choice := 0L]
# leave some tasks all-zero: outside option chosen
dt[, choice := if (runif(1) < 0.8) sample(c(1L, rep(0L, J - 1))) else 0L,
  by = task]
input <- prepare_hmnl_data(dt, "task", "alt", "choice", c("x1", "x2"),
  person_col = "pid",
  alt_covariate_cols = "quality")

str(input$Z)
input$alt_mapping
```

---

```
prepare_hmnp_data
```

---

*Prepare inputs for hierarchical multinomial probit estimation*

---

### Description

Prepares and validates panel (or cross-sectional) choice data for the hierarchical Bayesian multinomial probit with iid  $N(0, \sigma^2)$  utility shocks. The model shares its two-level random-effect structure with [prepare\\_hmnl\\_data\(\)](#): respondent-level structural tastes  $\beta_i \sim N(b, W)$  over the covariate\_cols (normal only — the probit keeps full conjugacy), and a global alternative-level effect  $\delta_j = z_j' \theta + \xi_j$ ,  $\xi_j \sim N(0, \sigma_d^2)$ .

**Usage**

```
prepare_hmnp_data(
  data,
  id_col,
  alt_col,
  choice_col,
  covariate_cols,
  person_col = NULL,
  alt_covariate_cols = NULL,
  outside_opt_label = NULL,
  cf_residual_col = NULL,
  include_outside_option = TRUE
)
```

**Arguments**

|                                     |                                                                                                                                                                                             |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code>                   | Data frame containing choice data.                                                                                                                                                          |
| <code>id_col</code>                 | Name of the column identifying choice situations (tasks). Task ids only need to be unique within a respondent.                                                                              |
| <code>alt_col</code>                | Name of the column identifying alternatives.                                                                                                                                                |
| <code>choice_col</code>             | Name of the column indicating the chosen alternative (1 = chosen, 0 = not chosen).                                                                                                          |
| <code>covariate_cols</code>         | Vector of names of structural covariate columns (the random-coefficient dimensions).                                                                                                        |
| <code>person_col</code>             | Name of the respondent column grouping choice situations. <code>NULL</code> (default) makes each choice situation its own respondent.                                                       |
| <code>alt_covariate_cols</code>     | Names of alternative-level covariate columns (constant within each alternative) forming the $\delta$ mean function. <code>NULL</code> (default) gives an intercept-only design ( $P = 1$ ). |
| <code>outside_opt_label</code>      | Label of physical outside-option rows, removed when <code>include_outside_option = TRUE</code> (the outside good is implicit).                                                              |
| <code>cf_residual_col</code>        | Name of a first-stage residual column (control function for an endogenous covariate), appended to <code>X</code> . Default <code>NULL</code> .                                              |
| <code>include_outside_option</code> | Logical; if <code>TRUE</code> (default) an implicit outside option with systematic utility 0 is part of every choice set.                                                                   |

**Details**

The returned structure is identical to `prepare_hmnl_data()` (both preps share one internal engine), except there is no `rc_dist` field. Unlike `prepare_mnp_data()`, utilities are NOT differenced against a base alternative: the iid-shock model works in un-differenced utility space, so unbalanced choice sets are supported and the outside option is implicit (its latent utility is a stochastic  $N(0, \sigma^2)$  draw in the kernel, systematic utility 0).

**Value**

A list of class `c("choicer_data_hmnp", "list")` with the same components as `prepare_hmnp_data()` (minus `rc_dist`).

**Column names**

No column the model uses (the `id`, `alternative`, `choice`, `covariate`, `weight`, `cluster`, `nest` or `decision-maker` column) may have a name starting with `.choicer_`, which `choicer` reserves for its working columns, and the `alternative` column may not be named `alt_int`, `N_OBS`, `N_CHOICES`, `TAKE_RATE` or `MKT_SHARE`, the names of the other columns of `alt_mapping`. Either is an error that names the column. Columns the model does not use may have any name.

**See Also**

`prepare_hmnp_data()` for the component-by-component description.

**Examples**

```
library(data.table)
set.seed(42)
N <- 20; T <- 3; J <- 4
dt <- data.table(
  pid = rep(1:N, each = T * J),
  task = rep(seq_len(N * T), each = J),
  alt = rep(1:J, N * T)
)
dt[, `:=`(x1 = rnorm(.N), x2 = runif(.N, -1, 1))]
dt[, choice := 0L]
dt[, choice := if (runif(1) < 0.8) sample(c(1L, rep(0L, J - 1))) else 0L,
  by = task]
input <- prepare_hmnp_data(dt, "task", "alt", "choice", c("x1", "x2"),
  person_col = "pid")
input$Ti[1:5]
input$alt_mapping
```

---

```
prepare_mnl_data
```

---

```
Prepare inputs for multinomial logit estimation
```

---

**Description**

Prepares and validates inputs for multinomial logit estimation routine.

**Usage**

```
prepare_mnl_data(
  data,
  id_col,
  alt_col,
```

```

    choice_col,
    covariate_cols,
    weights = NULL,
    outside_opt_label = NULL,
    include_outside_option = FALSE,
    weights_col = NULL,
    cluster_col = NULL
  )

```

## Arguments

|                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code>                   | Data frame containing choice data.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <code>id_col</code>                 | Name of the column identifying choice situations (individuals).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <code>alt_col</code>                | Name of the column identifying alternatives.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>choice_col</code>             | Name of the column indicating chosen alternative (1 = chosen, 0 = not chosen).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>covariate_cols</code>         | Vector of names of columns to be used as covariates.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <code>weights</code>                | Optional vector of weights for each choice situation. If <code>NULL</code> , equal weights are used. All weights must be finite and strictly positive.                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <code>outside_opt_label</code>      | The label of the outside option or of the base alternative, given in the class of the alternative column. With <code>include_outside_option = TRUE</code> it labels the outside option, which the model adds: rows of the data that carry it are dropped, and it names the outside option's row of <code>alt_mapping</code> . Without the outside option it names the base alternative, coded first (with constants, its constant is normalized to 0); it must be one of the alternatives, and <code>NULL</code> (the default) makes the first alternative in sort order the base. |
| <code>include_outside_option</code> | Logical indicating whether to include an outside option in the model.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <code>weights_col</code>            | Optional name of a column in data holding per-row weights. The column must be constant within each <code>id_col</code> (one weight per choice situation) and is collapsed accordingly. Mutually exclusive with <code>weights</code> . All weights must be finite and strictly positive.                                                                                                                                                                                                                                                                                            |
| <code>cluster_col</code>            | Optional name of a column in data holding cluster labels for cluster-robust standard errors (e.g. a person id when the same decision maker contributes several choice situations). Must be constant within each <code>id_col</code> . Collapsed to one label per choice situation and returned as <code>cluster</code> ; used by <code>se_method = "cluster"</code> and <code>vcov(fit, type = "cluster")</code> .                                                                                                                                                                 |

## Value

A list containing:

- `X`: Design matrix (sum(M) x K, double).
- `alt_idx`: Integer vector of alternative indices.
- `choice_idx`: Integer vector of chosen alternative indices.
- `M`: Integer vector with number of alternatives per choice situation.

- N: Number of choice situations.
- weights: Vector of weights.
- cluster: Vector of cluster labels (or NULL).
- situation\_ids: Choice-situation ids in prepared (sorted) order.
- include\_outside\_option: Logical flag.
- alt\_mapping: Data.table mapping alternatives to summary statistics.
- dropped\_cols: Names of columns dropped due to collinearity, if any.

### Column names

No column the model uses (the id, alternative, choice, covariate, weight, cluster, nest or decision-maker column) may have a name starting with .choicer\_, which choicer reserves for its working columns, and the alternative column may not be named alt\_int, N\_OBS, N\_CHOICES, TAKE\_RATE or MKT\_SHARE, the names of the other columns of alt\_mapping. Either is an error that names the column. Columns the model does not use may have any name.

### Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
input <- prepare_mnl_data(dt, "id", "alt", "choice", c("x1", "x2"))
str(input$X)
input$alt_mapping
```

---

```
prepare_mnp_data
```

---

```
Prepare inputs for Bayesian multinomial probit estimation
```

---

### Description

Prepares and validates inputs for Bayesian multinomial probit estimation. Covariates are differenced against the base alternative, so the design matrix has one row per (choice situation, non-base alternative) pair. Balanced choice sets are required: every choice situation must contain the same  $J$  alternatives.

### Usage

```
prepare_mnp_data(
  data,
  id_col,
  alt_col,
  choice_col,
```

```

    covariate_cols,
    base_alt = NULL,
    use_asc = TRUE
  )

```

### Arguments

|                             |                                                                                                                                                                                                                                                               |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code>           | Data frame containing choice data.                                                                                                                                                                                                                            |
| <code>id_col</code>         | Name of the column identifying choice situations (individuals).                                                                                                                                                                                               |
| <code>alt_col</code>        | Name of the column identifying alternatives.                                                                                                                                                                                                                  |
| <code>choice_col</code>     | Name of the column indicating chosen alternative (1 = chosen, 0 = not chosen).                                                                                                                                                                                |
| <code>covariate_cols</code> | Vector of names of columns to be used as covariates.                                                                                                                                                                                                          |
| <code>base_alt</code>       | Label of the base (reference) alternative used for utility differencing. If NULL (default), the first alternative in sort order is used.                                                                                                                      |
| <code>use_asc</code>        | Logical indicating whether to include alternative-specific constants (one intercept per non-base alternative), named <code>ASC_&lt;label&gt;</code> . No covariate may have one of these names; ASC dummies built by hand need <code>use_asc = FALSE</code> . |

### Value

A list containing:

- `X`: Stacked differenced design matrix  $((N * p) \times K)$ , double, covariate columns first, then ASC columns when `use_asc = TRUE`.
- `y`: Integer vector of choices (0 = base alternative,  $j$  in  $1..p$  for the  $j$ -th non-base alternative), one per choice situation.
- `p`: Number of utility differences ( $J - 1$ ).
- `J`: Number of alternatives.
- `N`: Number of choice situations.
- `K`: Number of columns of `X`.
- `alt_mapping`: `Data.table` mapping alternatives to summary statistics (the base alternative is `alt_int = 1`).
- `base_alt`: Resolved label of the base alternative.
- `param_map`: Named list of integer index vectors (`beta`, `asc`).
- `use_asc`: Logical flag.
- `dropped_cols`: Names of columns dropped due to collinearity, if any.
- `data_spec`: List with column name metadata.

### Column names

No column the model uses (the `id`, `alternative`, `choice`, `covariate`, `weight`, `cluster`, `nest` or `decision-maker` column) may have a name starting with `.choicer_`, which `choicer` reserves for its working columns, and the `alternative` column may not be named `alt_int`, `N_OBS`, `N_CHOICES`, `TAKE_RATE` or `MKT_SHARE`, the names of the other columns of `alt_mapping`. Either is an error that names the column. Columns the model does not use may have any name.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
input <- prepare_mnp_data(dt, "id", "alt", "choice", c("x1", "x2"))
str(input$X)
input$alt_mapping
```

---

|                  |                                                  |
|------------------|--------------------------------------------------|
| prepare_mxl_data | <i>Prepare inputs for mixed logit estimation</i> |
|------------------|--------------------------------------------------|

---

**Description**

Prepares and validates inputs for mixed logit estimation routine.

**Usage**

```
prepare_mxl_data(
  data,
  id_col,
  alt_col,
  choice_col,
  covariate_cols,
  random_var_cols,
  weights = NULL,
  outside_opt_label = NULL,
  include_outside_option = FALSE,
  rc_correlation = FALSE,
  weights_col = NULL,
  cluster_col = NULL,
  person_col = NULL
)
```

**Arguments**

|                 |                                                                               |
|-----------------|-------------------------------------------------------------------------------|
| data            | Data frame containing choice data                                             |
| id_col          | Name of the column identifying choice situations                              |
| alt_col         | Name of the column identifying alternatives                                   |
| choice_col      | Name of the column indicating chosen alternative (1 = chosen, 0 = not chosen) |
| covariate_cols  | Vector of names of columns to be used as covariates                           |
| random_var_cols | Vector of names of columns to be used as random variables                     |

|                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>weights</code>                | Optional vector of weights, one per choice situation in the prepared order (ascending id; see Details). If NULL, equal weights are used. All weights must be finite and strictly positive. With <code>person_col</code> they are decision-maker weights and must be constant within each decision maker.                                                                                                                                                                                                                                   |
| <code>outside_opt_label</code>      | The label of the outside option or of the base alternative, given in the class of the alternative column. With <code>include_outside_option = TRUE</code> it names the outside option's row of <code>alt_mapping</code> . Without the outside option it names the base alternative, coded first (with constants, its constant is normalized to 0); it must be one of the alternatives, and NULL (the default) makes the first alternative in sort order the base.                                                                          |
| <code>include_outside_option</code> | Logical indicating whether to include an outside option in the model.                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <code>rc_correlation</code>         | Logical indicating whether random coefficients are correlated. Default is FALSE.                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <code>weights_col</code>            | Optional name of a column in data holding a per-row weight (constant within each choice situation, and within each decision maker when <code>person_col</code> is used; finite and strictly positive). Mutually exclusive with <code>weights</code> .                                                                                                                                                                                                                                                                                      |
| <code>cluster_col</code>            | Optional name of a column in data holding cluster labels for cluster-robust standard errors. Must be constant within each <code>id_col</code> , and within each decision maker when <code>person_col</code> is used (clusters must nest decision makers); collapsed to one label per choice situation and returned as <code>cluster</code> .                                                                                                                                                                                               |
| <code>person_col</code>             | Optional name of the column identifying decision makers (respondents). When supplied, all choice situations of a decision maker share one draw of the random coefficients (panel likelihood); <code>id_col</code> must still identify choice situations uniquely across the data set. NULL (default) makes each choice situation its own decision maker (cross-sectional likelihood). Choice-based (WESML) samples are not supported with <code>person_col</code> : their weights vary with the chosen alternative, not by decision maker. |

## Details

Rows are ordered by choice-situation id and, within a situation, by alternative. With `person_col`, rows are ordered by decision maker first, then by situation id and alternative, so that each decision maker's situations are contiguous. A positional `weights` vector is read in the prepared order, which is ascending id in the cross-section; with `person_col` it is accepted only when the prepared order is also ascending id, and an error otherwise points to `weights_col`, which is aligned by id and is the safer interface.

## Value

A `choicer_data_mxl` object (list) containing:

- `X`: Fixed-coefficient design matrix ( $\text{sum}(M) \times K_x$ , double).
- `W`: Random-coefficient design matrix ( $\text{sum}(M) \times K_w$ , double).
- `alt_idx`: Integer vector of alternative indices.
- `choice_idx`: Integer vector of chosen alternative indices.
- `M`: Integer vector with number of alternatives per choice situation.

- N: Number of choice situations.
- weights: Vector of weights.
- cluster: Vector of cluster labels (or NULL).
- situation\_ids: Choice-situation ids in prepared (sorted) order.
- Ti: Number of choice situations of each decision maker, in prepared order (NULL without person\_col).
- person\_ids: Decision-maker ids in prepared order (NULL without person\_col).
- include\_outside\_option: Logical flag.
- rc\_correlation: Logical flag.
- alt\_mapping: data.table mapping alternatives to summary statistics.
- dropped\_cols: Names of columns dropped due to collinearity, if any.
- data\_spec: List with column-name metadata (incl. person\_col).

### Column names

No column the model uses (the id, alternative, choice, covariate, weight, cluster, nest or decision-maker column) may have a name starting with .choicer\_, which choicer reserves for its working columns, and the alternative column may not be named alt\_int, N\_OBS, N\_CHOICES, TAKE\_RATE or MKT\_SHARE, the names of the other columns of alt\_mapping. Either is an error that names the column. Columns the model does not use may have any name.

### Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N), w2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
input <- prepare_mxl_data(dt, "id", "alt", "choice", "x1", c("w1", "w2"))
str(input$X)
str(input$W)
```

---

```
prepare_nl_data
```

---

```
Prepare inputs for nested logit estimation
```

---

### Description

Validates inputs, builds design matrices, and constructs nest structure for nested logit estimation. Calls [prepare\\_mnl\\_data](#) internally for base data preparation, then adds nest-specific fields.

**Usage**

```
prepare_nl_data(
  data,
  id_col,
  alt_col,
  choice_col,
  covariate_cols,
  nest_col,
  weights = NULL,
  outside_opt_label = NULL,
  include_outside_option = FALSE,
  weights_col = NULL,
  cluster_col = NULL
)
```

**Arguments**

|                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code>                   | Data frame containing choice data.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <code>id_col</code>                 | Name of the column identifying choice situations (individuals).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <code>alt_col</code>                | Name of the column identifying alternatives.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>choice_col</code>             | Name of the column indicating chosen alternative (1 = chosen, 0 = not chosen).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>covariate_cols</code>         | Vector of names of columns to be used as covariates.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <code>nest_col</code>               | Name of the column mapping each alternative to its nest. Every alternative must belong to exactly one nest.                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <code>weights</code>                | Optional vector of weights for each choice situation. If <code>NULL</code> , equal weights are used. All weights must be finite and strictly positive.                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <code>outside_opt_label</code>      | The label of the outside option or of the base alternative, given in the class of the alternative column. With <code>include_outside_option = TRUE</code> it labels the outside option, which the model adds: rows of the data that carry it are dropped, and it names the outside option's row of <code>alt_mapping</code> . Without the outside option it names the base alternative, coded first (with constants, its constant is normalized to 0); it must be one of the alternatives, and <code>NULL</code> (the default) makes the first alternative in sort order the base. |
| <code>include_outside_option</code> | Logical indicating whether to include an outside option in the model.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <code>weights_col</code>            | Optional name of a column in data holding per-row weights. The column must be constant within each <code>id_col</code> (one weight per choice situation) and is collapsed accordingly. Mutually exclusive with <code>weights</code> . All weights must be finite and strictly positive.                                                                                                                                                                                                                                                                                            |
| <code>cluster_col</code>            | Optional name of a column in data holding cluster labels for cluster-robust standard errors. Must be constant within each <code>id_col</code> ; collapsed to one label per choice situation and returned as <code>cluster</code> .                                                                                                                                                                                                                                                                                                                                                 |

**Value**

A choicer\_data\_n1 object (list) containing:

- All fields from `prepare_mnl_data` (`X`, `alt_idx`, `choice_idx`, `M`, `N`, `weights`, `cluster`, `situation_ids`, `include_outside_option`, `alt_mapping`, `dropped_cols`).
- `nest_idx`: Integer vector of length `J` mapping each alternative (in `alt_mapping` row order) to its nest.
- `data_spec`: List with column name metadata including `nest_col`.

**Column names**

No column the model uses (the `id`, `alternative`, `choice`, `covariate`, `weight`, `cluster`, `nest` or decision-maker column) may have a name starting with `.choicer_`, which choicer reserves for its working columns, and the alternative column may not be named `alt_int`, `N_OBS`, `N_CHOICES`, `TAKE_RATE` or `MKT_SHARE`, the names of the other columns of `alt_mapping`. Either is an error that names the column. Columns the model does not use may have any name.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, nest := ifelse(alt <= 2, "A", "B")]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
input <- prepare_n1_data(dt, "id", "alt", "choice", c("x1", "x2"), "nest")
input$nest_idx
input$alt_mapping
```

---

|                  |                                         |
|------------------|-----------------------------------------|
| print.choicer_cs | <i>Print a consumer surplus summary</i> |
|------------------|-----------------------------------------|

---

**Description**

Print a consumer surplus summary

**Usage**

```
## S3 method for class 'choicer_cs'
print(x, digits = 4, ...)
```

**Arguments**

|                     |                                        |
|---------------------|----------------------------------------|
| <code>x</code>      | A choicer_cs object.                   |
| <code>digits</code> | Number of significant digits to print. |
| <code>...</code>    | Additional arguments (ignored).        |

**Value**

The object invisibly.

---

|                   |                                   |
|-------------------|-----------------------------------|
| print.choicer_fit | <i>Print a choicer_fit object</i> |
|-------------------|-----------------------------------|

---

**Description**

Prints a brief summary of the fitted model.

**Usage**

```
## S3 method for class 'choicer_fit'
print(x, ...)
```

**Arguments**

|     |                                 |
|-----|---------------------------------|
| x   | A choicer_fit object.           |
| ... | Additional arguments (ignored). |

**Value**

The object invisibly.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
print(fit)
```

---

|                   |                                       |
|-------------------|---------------------------------------|
| print.choicer_gof | <i>Print goodness-of-fit measures</i> |
|-------------------|---------------------------------------|

---

**Description**

Print goodness-of-fit measures

**Usage**

```
## S3 method for class 'choicer_gof'  
print(x, ...)
```

**Arguments**

|     |                                 |
|-----|---------------------------------|
| x   | A choicer_gof object.           |
| ... | Additional arguments (ignored). |

**Value**

The object invisibly.

---

|                  |                                       |
|------------------|---------------------------------------|
| print.choicer_hb | <i>Print a hierarchical Bayes fit</i> |
|------------------|---------------------------------------|

---

**Description**

Print a hierarchical Bayes fit

**Usage**

```
## S3 method for class 'choicer_hb'  
print(x, ...)
```

**Arguments**

|     |                                        |
|-----|----------------------------------------|
| x   | A choicer_hmnl or choicer_hmnp object. |
| ... | Additional arguments (ignored).        |

**Value**

The object invisibly.

**Examples**

```
sim <- simulate_hmnl_data(N = 50, T = 2, J = 3, seed = 42)
fit <- suppressWarnings(run_hmnllogit(sim$data, "task", "alt", "choice", c("x1", "x2"),
                                   person_col = "pid",
                                   mcmc = list(R = 300, burn = 100)))
print(fit)
```

---

```
print.choicer_mnp      Print a choicer_mnp object
```

---

**Description**

Prints a brief summary of the fitted Bayesian multinomial probit model.

**Usage**

```
## S3 method for class 'choicer_mnp'
print(x, ...)
```

**Arguments**

```
x          A choicer_mnp object.
...        Additional arguments (ignored).
```

**Value**

The object invisibly.

**Examples**

```
library(data.table)
set.seed(42)
N <- 100; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnpobit(dt, "id", "alt", "choice", c("x1", "x2"),
                  mcmc = list(R = 300, burn = 100))
print(fit)
```

---

```
print.choicer_tastes
```

*Print conditional tastes*

---

### Description

Print conditional tastes

### Usage

```
## S3 method for class 'choicer_tastes'
print(x, digits = 3, ...)
```

### Arguments

|        |                                                                   |
|--------|-------------------------------------------------------------------|
| x      | A choicer_tastes object from <a href="#">conditional_tastes</a> . |
| digits | Number of decimal places for the population table.                |
| ...    | Additional arguments (ignored).                                   |

### Value

The object invisibly.

---

```
print.choicer_wtp
```

*Print a WTP table*

---

### Description

Print a WTP table

### Usage

```
## S3 method for class 'choicer_wtp'
print(x, digits = 4, ...)
```

### Arguments

|        |                                                                |
|--------|----------------------------------------------------------------|
| x      | A choicer_wtp object.                                          |
| digits | Number of significant digits to print.                         |
| ...    | Additional arguments passed to <code>print.data.frame</code> . |

### Value

The object invisibly.

---

```
print.summary.choicer_hb
```

*Print the summary of a hierarchical Bayes fit*

---

**Description**

Print the summary of a hierarchical Bayes fit

**Usage**

```
## S3 method for class 'summary.choicer_hb'  
print(x, ...)
```

**Arguments**

|     |                                 |
|-----|---------------------------------|
| x   | A summary.choicer_hb object.    |
| ... | Additional arguments (ignored). |

**Value**

The object invisibly.

---

```
print.summary.choicer_mnl
```

*Print summary for multinomial logit model*

---

**Description**

Print summary for multinomial logit model

**Usage**

```
## S3 method for class 'summary.choicer_mnl'  
print(x, ...)
```

**Arguments**

|     |                                 |
|-----|---------------------------------|
| x   | A summary.choicer_mnl object.   |
| ... | Additional arguments (ignored). |

**Value**

The object invisibly.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
print(summary(fit))
```

---

```
print.summary.choicer_mnp
```

*Print summary for Bayesian multinomial probit model*

---

**Description**

Print summary for Bayesian multinomial probit model

**Usage**

```
## S3 method for class 'summary.choicer_mnp'
print(x, ...)
```

**Arguments**

```
x          A summary.choicer_mnp object.
...        Additional arguments (ignored).
```

**Value**

The object invisibly.

**Examples**

```
library(data.table)
set.seed(42)
N <- 100; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnpbprobit(dt, "id", "alt", "choice", c("x1", "x2"),
                     mcmc = list(R = 300, burn = 100))
print(summary(fit))
```

---

```
print.summary.choicer_mxl
```

*Print summary for mixed logit model*

---

**Description**

Print summary for mixed logit model

**Usage**

```
## S3 method for class 'summary.choicer_mxl'
print(x, ...)
```

**Arguments**

`x`                    A summary.choicer\_mxl object.  
`...`                Additional arguments (ignored).

**Value**

The object invisibly.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mxlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = "x1", random_var_cols = "w1", S = 50L
)
print(summary(fit))
```

---

```
print.summary.choicer_nl
```

*Print summary for nested logit model*

---

**Description**

Print summary for nested logit model

**Usage**

```
## S3 method for class 'summary.choicer_nl'
print(x, ...)
```

**Arguments**

**x**                      A summary.choicer\_nl object.

**...**                  Additional arguments (ignored).

**Value**

The object invisibly.

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, nest := ifelse(alt <= 2, "A", "B")]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_nestlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = c("x1", "x2"), nest_col = "nest"
)
print(summary(fit))
```

---

|                |                                 |
|----------------|---------------------------------|
| recovery_table | <i>Parameter recovery table</i> |
|----------------|---------------------------------|

---

**Description**

Compares fitted coefficients to a set of true parameter values on the same scale as the estimator's internal parameterization. Returns one row per estimated parameter with true value, estimate, standard error, bias, relative bias (%), z-score against the truth, Wald CI, and a coverage indicator.

**Usage**

```
recovery_table(object, truth = NULL, level = 0.95, ...)

## S3 method for class 'choicer_fit'
recovery_table(object, truth = NULL, level = 0.95, ...)

## S3 method for class 'choicer_mnp'
recovery_table(object, truth = NULL, level = 0.95, ...)
```

```
## S3 method for class 'choicer_mc'
recovery_table(object, truth = NULL, level = 0.95, ...)

## S3 method for class 'choicer_hb'
recovery_table(object, truth = NULL, level = 0.95, ...)
```

### Arguments

|                     |                                                                                                                                    |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------|
| <code>object</code> | A <code>choicer_fit</code> object (MNL, MXL, or NL) or a <code>choicer_mc</code> result.                                           |
| <code>truth</code>  | Either a <code>choicer_sim</code> object (whose <code>\$true_params</code> will be used) or a named list of true parameter values. |
| <code>level</code>  | Confidence level for the Wald CI and coverage indicator. Default 0.95.                                                             |
| <code>...</code>    | Unused.                                                                                                                            |

### Details

For MXL fits the sigma block compares the raw Cholesky parameters (`L_params`), not the re-constructed covariance matrix. For log-normal random-coefficient means the raw  $\mu$  estimate is compared directly; callers who want recovery on the DGP scale ( $\exp(\mu)$ ) should transform both sides before calling.

When the estimator has normalized the first inside alternative's ASC to zero (which happens for MNL/MXL with `include_outside_option = FALSE` and no outside option baked into the fit), the first entry of `truth$delta` is dropped before the comparison so lengths match.

### Value

See class-specific methods.

### Methods (by class)

- `recovery_table(choicer_fit)`: Returns a `choicer_recovery` object (a `data.table`) with columns `parameter`, `group`, `true`, `estimate`, `se`, `bias`, `rel_bias_pct`, `z_vs_true`, `lower_ci`, `upper_ci`, `covers`.
- `recovery_table(choicer_mnp)`: Method for Bayesian MNP fits (`choicer_mnp`). The `estimate` column holds posterior means of the identified draws and `se` holds their posterior standard deviations, so `lower_ci` / `upper_ci` are normal-approximation credible intervals. In addition to the `beta` and `asc` blocks, a `sigma` block compares the identified covariance of the utility differences (lower triangle in the estimator's row-major `Sigma_ij` order); its first row is the `sigma_11 = 1` normalization and is exact by construction. `truth` must be on the identified scale, as returned by `simulate_mnp_data()`.
- `recovery_table(choicer_mc)`: For a `choicer_mc` object, delegates to `summary(object, level)` and returns a `choicer_mc_summary`. Inspect `object$replications` directly for per-rep detail.
- `recovery_table(choicer_hb)`: Method for hierarchical Bayes fits (`choicer_hmnl` / `choicer_hmnp`). `estimate` holds posterior means and `se` posterior standard deviations, so `lower_ci` / `upper_ci` are normal-approximation credible intervals. Blocks: `beta` (population means `b` vs `truth$beta`),

w (diag(W) vs diag(truth\$W)), theta (delta mean function), sigma\_d (the SD, compared through the sqrt of the sigma\_d^2 draws), and delta (per-alternative effects vs the realized truth\$delta). For an HMNP fit, truth must be on the identified scale, as returned by `simulate_hmnp_data()`.

## Examples

```
sim <- simulate_mnl_data(N = 2000, J = 4, seed = 123)
fit <- run_mnlogit(
  data = sim$data, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = c("x1", "x2"),
  outside_opt_label = 0L, include_outside_option = FALSE, use_asc = TRUE
)
recovery_table(fit, sim)
```

---

|      |                                                          |
|------|----------------------------------------------------------|
| rhat | <i>Split-<math>\hat{R}</math> convergence diagnostic</i> |
|------|----------------------------------------------------------|

---

## Description

Computes the split- $\hat{R}$  (potential scale reduction factor) of Gelman et al. for each column of a matrix of posterior draws. Every chain is split in half, so the diagnostic detects non-stationarity within a single chain as well as disagreement across chains; values near 1 indicate convergence, and values above roughly 1.05 warrant a longer run.

## Usage

```
rhat(draws, rank = FALSE)
```

## Arguments

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| draws | A matrix of posterior draws (rows = iterations, columns = parameters) for a single chain, or a list of such matrices (one per chain, identical dimensions).                                                                                                                                                                                                                                                                                                            |
| rank  | Logical; if FALSE (default) computes the classic split $\hat{R}$ (unchanged from prior releases, bit-for-bit). If TRUE, computes the rank-normalized, folded $\hat{R}$ of Vehtari, Gelman, Simpson, Carpenter & Buerkner (2021, <i>Bayesian Analysis</i> ): the max of a bulk (rank-normalized draws) and a fold (rank-normalized absolute deviation from the pooled median) split- $R-hat$ , which is more robust to heavy tails and scale differences across chains. |

## Value

Named numeric vector with one  $\hat{R}$  per parameter (NA for parameters with zero variance).

## Examples

```
set.seed(42)
draws <- matrix(rnorm(2000), ncol = 2,
                dimnames = list(NULL, c("a", "b")))
rhat(draws)          # ~1: white noise is stationary
drifting <- cbind(a = cumsum(rnorm(1000)))
rhat(drifting)       # >> 1: a random walk is not
rhat(draws, rank = TRUE) # rank-normalized variant
```

---

run\_hmnlogit

*Fit a hierarchical Bayesian multinomial logit (HMNL)*


---

## Description

Runs the adaptive RW-Metropolis-within-Gibbs sampler for the hierarchical (random-coefficients, panel or cross-sectional) multinomial logit with a BLP-style alternative-level random effect:

$$U_{ijt} = x'_{ijt}\gamma_i + \delta_j + \epsilon_{ijt}, \quad U_{iot} = \epsilon_{iot},$$

with i.i.d. Gumbel shocks (including on the implicit outside option, whose systematic utility is 0),  $\beta_i \sim N(b, W)$  over the structural covariates ( $\gamma_{ik} = \beta_{ik}$  or  $\exp(\beta_{ik})$  per `rc_dist`), and  $\delta_j = z'_j\theta + \xi_j$ ,  $\xi_j \sim N(0, \sigma_d^2)$ . Partial pooling shrinks each  $\delta_j$  toward its characteristics-based mean  $z'_j\theta$ ; the outside option anchors the level of  $\delta$  (mean utility relative to the outside good), so no base alternative or sum-to-zero constraint is needed.

## Usage

```
run_hmnlogit(
  data = NULL,
  id_col = NULL,
  alt_col = NULL,
  choice_col = NULL,
  covariate_cols = NULL,
  person_col = NULL,
  alt_covariate_cols = NULL,
  outside_opt_label = NULL,
  cf_residual_col = NULL,
  input_data = NULL,
  include_outside_option = TRUE,
  rc_dist = NULL,
  prior = list(),
  mcmc = list(),
  chains = 1,
  keep_beta_i = c("means", "draws", "none"),
  keep_data = TRUE
)
```

**Arguments**

|                                     |                                                                                                                                                                                                                                                                                                                                                              |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code>                   | Data frame (convenience pathway). Supply either <code>data</code> (with the column names) or <code>input_data</code> , not both.                                                                                                                                                                                                                             |
| <code>id_col</code>                 | Name of the column identifying choice situations (tasks). Task ids only need to be unique within a respondent.                                                                                                                                                                                                                                               |
| <code>alt_col</code>                | Name of the column identifying alternatives.                                                                                                                                                                                                                                                                                                                 |
| <code>choice_col</code>             | Name of the column indicating the chosen alternative (1 = chosen, 0 = not chosen).                                                                                                                                                                                                                                                                           |
| <code>covariate_cols</code>         | Vector of names of structural covariate columns (the random-coefficient dimensions).                                                                                                                                                                                                                                                                         |
| <code>person_col</code>             | Name of the respondent column grouping choice situations. NULL (default) makes each choice situation its own respondent.                                                                                                                                                                                                                                     |
| <code>alt_covariate_cols</code>     | Names of alternative-level covariate columns (constant within each alternative) forming the $\delta$ mean function. NULL (default) gives an intercept-only design ( $P = 1$ ).                                                                                                                                                                               |
| <code>outside_opt_label</code>      | Label of physical outside-option rows, removed when <code>include_outside_option = TRUE</code> (the outside good is implicit).                                                                                                                                                                                                                               |
| <code>cf_residual_col</code>        | Name of a first-stage residual column (control function for an endogenous covariate), appended to $X$ . Default NULL.                                                                                                                                                                                                                                        |
| <code>input_data</code>             | A <code>choicer_data_hmnl</code> object from <code>prepare_hmnl_data()</code> (advanced pathway).                                                                                                                                                                                                                                                            |
| <code>include_outside_option</code> | Logical; if TRUE (default) an implicit outside option with systematic utility 0 is part of every choice set.                                                                                                                                                                                                                                                 |
| <code>rc_dist</code>                | Integer vector, one entry per column of <code>covariate_cols</code> : 0 for a normal random coefficient, 1 for log-normal (the coefficient enters utility as $\exp(\beta_{ik})$ ; hierarchy normal on the log scale). Default NULL is all-normal. Automatically aligned through dropped columns; a <code>cf_residual_col</code> coordinate is always normal. |
| <code>prior</code>                  | Named list overriding prior defaults: <code>b_bar</code> (0), <code>A</code> (0.01 I), <code>nu</code> ( $K + 3$ ), <code>V</code> ( <code>nu</code> I), <code>theta_bar</code> (0), <code>A_theta</code> (0.01 I), <code>sd_prior</code> ( <code>list(half_cauchy = TRUE, s_d = 1, c0 = 3, d0 = 3)</code> ).                                                |
| <code>mcmc</code>                   | Named list overriding MCMC defaults: <code>R</code> (10000), <code>burn</code> ( <code>R</code> % 5), <code>thin</code> (1), <code>seed</code> (drawn via <code>sample.int()</code> so <code>set.seed()</code> governs), <code>trace</code> (0), <code>s_init</code> ( $2.38 / \sqrt{K}$ ), <code>accept_target</code> (0.234).                              |
| <code>chains</code>                 | Number of independent chains (seeds offset by 1, run sequentially). Chain 1 provides the reported draws; all chains feed the rank-normalized split-R-hat table and the retained chains field (all per-chain <code>b/w_vech/delta/theta/sigma_d2/loglik</code> draws, consumed by <code>ess()</code> , <code>mcse()</code> , and <code>traceplot()</code> ).  |
| <code>keep_beta_i</code>            | "means" (default) stores posterior means/SDs of the individual-level $\beta_i$ ; "draws" additionally stores the full ( $K, N, R_{\text{keep}}$ ) draw cube (memory-guarded, budgeted per chain); "none" stores neither.                                                                                                                                     |

`keep_data` Logical; keep the prepared data on the fit (default TRUE, needed by post-estimation methods).

## Details

**Initialization.**  $\beta_i$  start at the pooled MNL maximum likelihood estimate over the structural covariates (log-normal coordinates transformed to the chain scale with a warn-and-clamp at 0.05);  $\delta$  starts at shrunk log choice-share contrasts against the outside option;  $\theta$  at the OLS regression of the initial  $\delta$  on  $Z$ .

**Priors.**  $b \sim N(b\_bar, A^{-1})$ ,  $W \sim IW(\nu, V)$ ,  $\theta \sim N(\theta\_bar, A_\theta^{-1})$ , and  $\sigma_d \sim \text{half-Cauchy}(0, s_d)$  via the Makalic-Schmidt scale mixture (set `sd_prior$half_cauchy = FALSE` for a plain  $IG(c_0, d_0)$  on  $\sigma_d^2$ ).

**Endogeneity.** If a price-like covariate is endogenous (correlated with  $\xi_j$ ), supply a first-stage residual via `cf_residual_col` (Petrin & Train 2010); posterior uncertainty does NOT propagate first-stage estimation error.

For `beta_i` draws at very large scale beyond the memory guard's threshold, a future disk-streaming path (writing each kept slice to disk instead of retaining it in memory) is on the roadmap but not built in this phase; users needing per-respondent draws at that scale should reduce `R`, reduce chains, or use `keep_beta_i = "means"`.

## Value

A `choicer_hmnl` object (classed `c("choicer_hmnl", "choicer_hb")`) with posterior summaries (coefficients, se, vcov for  $b$ ; `theta_summary`; `sigma_d2_summary`; `W_mean`; `delta` and `xi` quality-ladder tables; `beta_i`), the raw thinned draws (chain 1), acceptance diagnostics in `accept`, the rank-normalized split-R-hat table in `rhat`, all chains' retained draws in `chains`, and sampler metadata.

## Column names

No column the model uses (the id, alternative, choice, covariate, weight, cluster, nest or decision-maker column) may have a name starting with `.choicer_`, which choicer reserves for its working columns, and the alternative column may not be named `alt_int`, `N_OBS`, `N_CHOICES`, `TAKE_RATE` or `MKT_SHARE`, the names of the other columns of `alt_mapping`. Either is an error that names the column. Columns the model does not use may have any name.

## See Also

[prepare\\_hmnl\\_data\(\)](#), [simulate\\_hmnl\\_data\(\)](#), [recovery\\_table\(\)](#), [rhat\(\)](#), [ess\(\)](#), [mcse\(\)](#), [traceplot\(\)](#)

## Examples

```
sim <- simulate_hmnl_data(N = 100, T = 3, J = 4, seed = 42)
fit <- suppressWarnings(run_hmnlogit(sim$data, "task", "alt", "choice", c("x1", "x2"),
  person_col = "pid", alt_covariate_cols = "z1",
  mcmc = list(R = 500, burn = 200))
summary(fit)
coef(fit, component = "delta")
```

run\_hmnprobit

*Fit a hierarchical Bayesian multinomial probit (HMNP)***Description**

Runs the fully conjugate Albert-Chib Gibbs sampler for the hierarchical multinomial probit with iid normal utility shocks in un-differenced utility space:

$$U_{ijt} = x'_{ijt}\beta_i + \delta_j + \epsilon_{ijt}, \quad U_{iot} = \epsilon_{iot}, \quad \epsilon \sim N(0, \sigma^2),$$

choice by argmax within the task including the stochastic implicit outside option,  $\beta_i \sim N(b, W)$  (normal coordinates only — log-normal would break conjugacy), and  $\delta_j = z'_j\theta + \xi_j$ ,  $\xi_j \sim N(0, \sigma_d^2)$ .

**Usage**

```
run_hmnprobit(
  data = NULL,
  id_col = NULL,
  alt_col = NULL,
  choice_col = NULL,
  covariate_cols = NULL,
  person_col = NULL,
  alt_covariate_cols = NULL,
  outside_opt_label = NULL,
  cf_residual_col = NULL,
  input_data = NULL,
  include_outside_option = TRUE,
  prior = list(),
  mcmc = list(),
  chains = 1,
  keep_beta_i = c("means", "draws", "none"),
  keep_data = TRUE
)
```

**Arguments**

|                |                                                                                                                |
|----------------|----------------------------------------------------------------------------------------------------------------|
| data           | Data frame (convenience pathway). Supply either data (with the column names) or input_data, not both.          |
| id_col         | Name of the column identifying choice situations (tasks). Task ids only need to be unique within a respondent. |
| alt_col        | Name of the column identifying alternatives.                                                                   |
| choice_col     | Name of the column indicating the chosen alternative (1 = chosen, 0 = not chosen).                             |
| covariate_cols | Vector of names of structural covariate columns (the random-coefficient dimensions).                           |

|                        |                                                                                                                                                                                                                                                                                                                                                                 |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| person_col             | Name of the respondent column grouping choice situations. NULL (default) makes each choice situation its own respondent.                                                                                                                                                                                                                                        |
| alt_covariate_cols     | Names of alternative-level covariate columns (constant within each alternative) forming the $\delta$ mean function. NULL (default) gives an intercept-only design ( $P = 1$ ).                                                                                                                                                                                  |
| outside_opt_label      | Label of physical outside-option rows, removed when include_outside_option = TRUE (the outside good is implicit).                                                                                                                                                                                                                                               |
| cf_residual_col        | Name of a first-stage residual column (control function for an endogenous covariate), appended to X. Default NULL.                                                                                                                                                                                                                                              |
| input_data             | A choicer_data_hmnp object from <a href="#">prepare_hmnp_data()</a> .                                                                                                                                                                                                                                                                                           |
| include_outside_option | Logical; if TRUE (default) an implicit outside option with systematic utility 0 is part of every choice set.                                                                                                                                                                                                                                                    |
| prior                  | As in <a href="#">run_hmnllogit()</a> , plus $a_0$ (3) and $s_0$ (3), the inverse-gamma shape/scale on the non-identified $\sigma^2$ .                                                                                                                                                                                                                          |
| mcmc                   | Named list overriding MCMC defaults: R (10000), burn (R %% 5), thin (1), seed (drawn via <code>sample.int()</code> so <code>set.seed()</code> governs), trace (0). No proposal-scale settings: the sampler is fully conjugate, with no Metropolis steps.                                                                                                        |
| chains                 | Number of independent chains (seeds offset by 1, run sequentially). Chain 1 provides the reported draws; all chains feed the rank-normalized split-R-hat table and the retained chains field (all per-chain $b/w\_vech/\delta/\theta/\sigma_d^2/\loglik$ draws, consumed by <a href="#">ess()</a> , <a href="#">mcse()</a> , and <a href="#">traceplot()</a> ). |
| keep_beta_i            | "means" (default) stores posterior means/SDs of the individual-level $\beta_i$ ; "draws" additionally stores the full (K, N, R_keep) draw cube (memory-guarded, budgeted per chain); "none" stores neither.                                                                                                                                                     |
| keep_data              | Logical; keep the prepared data on the fit (default TRUE, needed by post-estimation methods).                                                                                                                                                                                                                                                                   |

## Details

**Identification.** The probit likelihood is invariant to a common rescaling of utilities and  $\sigma$ , so the chain runs on the non-identified parameterization (free  $\sigma^2$ , better mixing via parameter expansion) and every kept draw is normalized by the matching power of the CURRENT  $\sigma$ : reported  $b/\sigma$ ,  $W/\sigma^2$ ,  $\delta/\sigma$ ,  $\theta/\sigma$ ,  $\sigma_d^2/\sigma^2$ . Raw chains are kept in `draws$*_raw`. The outside option anchors the location of  $\delta$  exactly as in [run\\_hmnllogit\(\)](#).

For `beta_i` draws at very large scale beyond the memory guard's threshold, a future disk-streaming path (writing each kept slice to disk instead of retaining it in memory) is on the roadmap but not built in this phase; users needing per-respondent draws at that scale should reduce R, reduce `chains`, or use `keep_beta_i = "means"`.

**Value**

A choicer\_hmnp object (classed c("choicer\_hmnp", "choicer\_hb")); the same layout as `run_hmnlogit()`'s return, with all reported summaries on the identified scale, raw chains in `draws$*_raw`, the non-identified `draws$sigma2` trace, and all chains' retained draws (identified scale) in `chains`.

**Column names**

No column the model uses (the id, alternative, choice, covariate, weight, cluster, nest or decision-maker column) may have a name starting with `.choicer_`, which choicer reserves for its working columns, and the alternative column may not be named `alt_int`, `N_OBS`, `N_CHOICES`, `TAKE_RATE` or `MKT_SHARE`, the names of the other columns of `alt_mapping`. Either is an error that names the column. Columns the model does not use may have any name.

**See Also**

`prepare_hmnp_data()`, `simulate_hmnp_data()`, `run_hmnlogit()`, `ess()`, `mcse()`, `traceplot()`

**Examples**

```
sim <- simulate_hmnp_data(N = 100, T = 3, J = 4, seed = 42)
fit <- suppressWarnings(run_hmnprobit(sim$data, "task", "alt", "choice", c("x1", "x2"),
  person_col = "pid", alt_covariate_cols = "z1",
  mcmc = list(R = 500, burn = 200)))
summary(fit)
```

---

run\_mnlogit

---

*Runs multinomial logit estimation*


---

**Description**

Estimates a multinomial logit model via maximum likelihood.

**Usage**

```
run_mnlogit(
  data = NULL,
  id_col = NULL,
  alt_col = NULL,
  choice_col = NULL,
  covariate_cols = NULL,
  input_data = NULL,
  optimizer = NULL,
  control = list(),
  weights = NULL,
  weights_col = NULL,
  outside_opt_label = NULL,
```

```

include_outside_option = FALSE,
use_asc = TRUE,
keep_data = TRUE,
scale_vars = c("bhhh", "none", "sd", "mad", "iqr"),
se_method = c("hessian", "bhhh", "sandwich", "cluster"),
cluster_col = NULL,
nloptr_opts = NULL
)

```

## Arguments

|                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code>              | Data frame containing choice data (convenience workflow). Mutually exclusive with <code>input_data</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>id_col</code>            | Name of the column identifying choice situations (individuals).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <code>alt_col</code>           | Name of the column identifying alternatives.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <code>choice_col</code>        | Name of the column indicating chosen alternative (1 = chosen, 0 = not chosen).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <code>covariate_cols</code>    | Vector of names of columns to be used as covariates. With <code>use_asc = TRUE</code> , none may be named like a constant, <code>ASC_&lt;label&gt;</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>input_data</code>        | List output from <a href="#">prepare_mnl_data</a> (advanced workflow). Mutually exclusive with <code>data</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <code>optimizer</code>         | Optimizer to use: "nloptr" (default), "optim", or a custom function with signature <code>f(theta_init, eval_f, lower, upper, control)</code> where <code>eval_f(theta)</code> returns <code>list(objective, gradient)</code> . Must return a list with <code>par/value</code> (or <code>solution/objective</code> ). If the custom function accepts <code>control</code> or <code>...</code> , the <code>control</code> argument is forwarded; otherwise it is silently ignored. With <code>scale_vars</code> other than "none", it receives <code>theta_init</code> , <code>lower</code> , <code>upper</code> and <code>eval_f</code> in the scaled coordinates, and its <code>par</code> is mapped back. |
| <code>control</code>           | List of optimizer-specific control parameters passed to the chosen optimizer (e.g., <code>list(maxeval = 2000)</code> for <code>nloptr</code> ). With <code>scale_vars</code> other than "none", settings in parameter units apply to the scaled coordinates.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <code>weights</code>           | Optional vector of weights for each choice situation. If <code>NULL</code> , equal weights are used. All weights must be finite and strictly positive.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>weights_col</code>       | Optional name of a column in <code>data</code> holding per-row weights (convenience workflow only). The column must be constant within each <code>id_col</code> (one weight per choice situation) and is collapsed accordingly. Mutually exclusive with <code>weights</code> . All weights must be finite and strictly positive. Used for choice-based / WESML weighting; pair with <code>se_method = "sandwich"</code> for valid inference.                                                                                                                                                                                                                                                               |
| <code>outside_opt_label</code> | The label of the outside option or of the base alternative, given in the class of the alternative column. With <code>include_outside_option = TRUE</code> it labels the outside option, which the model adds: rows of the data that carry it are dropped, and it names the outside option's row of <code>alt_mapping</code> . Without the outside option it names the base alternative, coded first (with constants, its constant is normalized to 0); it must be one of the alternatives, and <code>NULL</code> (the default) makes the first alternative in sort order the base.                                                                                                                         |

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| include_outside_option | Logical indicating whether to include an outside option in the model.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| use_asc                | Logical indicating whether to include alternative-specific constants (ASCs) in the model. An alternative's constant has no finite maximum-likelihood estimate when the alternative is never chosen in the data, or is chosen in every choice situation that offers it (the likelihood keeps rising as the constant falls, or grows); when the outside option or the base alternative is never chosen, or chosen wherever it is offered, the other constants have none together. The fit then warns (class <code>choicer_unbounded_constants</code> ) before the optimizer runs, naming them.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| keep_data              | Logical. If TRUE (default), stores prepared data in the returned object for <code>predict()</code> and post-estimation functions.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| scale_vars             | How the optimizer's coordinates are scaled. The choice does not change the estimator; it changes how fast, and whether, the optimizer reaches the maximum. "bhhh" (default) scales each parameter by the inverse square root of the diagonal of the BHHH (outer product of gradients) matrix at the start values (all zeros), rounded to a power of two, from one extra gradient pass, so that the BHHH diagonal, an estimate of the curvature, is near one at the start values in every coordinate, the constants' included (theirs follow the alternatives' shares). Where the diagonal is unusable (no finite, positive entry), the parameters are left unscaled, with a message. "none" hands the optimizer the parameters as they are (the default in <code>choicer</code> 0.2.x). With "sd" (sample standard deviation), "mad" ( <code>stats::mad</code> ) or "iqr" ( <code>stats::IQR(x) / 1.349</code> ), the optimizer works on the coefficients $X$ would have with every column divided by the chosen scale (each coefficient times its column's scale); the constants keep their scale, the data are not divided, and the fit holds no scaled copy of them. With any choice but "none" (so by default), the optimizer, a custom one included, works in the scaled coordinates, so settings in parameter units in <code>control</code> ( <code>nloptr</code> 's <code>xtol_abs</code> , <code>optim</code> 's <code>parscale</code> ) apply to the scaled coordinates; "none" keeps the parameters' units. Coefficients are reported in natural units and the standard errors computed in them, from the information matrix equilibrated before it is inverted, whatever the choice, so, where the likelihood is well identified and every choice reaches its maximum, reported quantities do not depend on this choice beyond the optimizer's tolerance. |
| se_method              | Method for computing standard errors: "hessian" (default, analytical Hessian), "bhhh" (outer product of gradients), "sandwich" (robust Huber-White / WESML variance $A^{-1}BA^{-1}$ ), or "cluster" (cluster-robust sandwich; requires <code>cluster_col</code> or a prepared <code>input_data</code> with a cluster field). Use "sandwich" under choice-based / WESML weighting. Any of these can also be recomputed post hoc via <code>vcov(fit, type = )</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| cluster_col            | Optional name of a column in data holding cluster labels for cluster-robust standard errors (e.g. a person id when the same decision maker contributes several choice situations). Must be constant within each <code>id_col</code> . Supplying <code>cluster_col</code> without an explicit <code>se_method</code> selects <code>se_method = "cluster"</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| nloptr_opts            | Deprecated. Use <code>optimizer</code> and <code>control</code> instead.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

## Details

Two workflows are supported:

**Convenience (default)** Supply data and column names. Data preparation (`prepare_mnl_data`) is handled automatically.

**Advanced** Call `prepare_mnl_data` yourself and pass the result via `input_data`.

## Value

A `choicer_mnl` object (inherits from `choicer_fit`). Standard S3 methods available: `summary()`, `coef()`, `vcov()`, `logLik()`, `AIC()`, `BIC()`, `nobs()`, `predict()`. The variance is computed once the optimizer has stopped. If that step fails with an error (an allocation in R, say), the fit returns its estimates without it (`vcov` `NULL`, `NA` standard errors, the error's message in `variance_error`) and warns, naming the `vcov(fit, type = )` that computes the variance later from a fit that keeps its data (`keep_data = TRUE`). An allocation that fails inside the compiled Hessian or BHHH loops still terminates R.

## Column names

No column the model uses (the `id`, `alternative`, `choice`, `covariate`, `weight`, `cluster`, `nest` or `decision-maker` column) may have a name starting with `.choicer_`, which `choicer` reserves for its working columns, and the `alternative` column may not be named `alt_int`, `N_OBS`, `N_CHOICES`, `TAKE_RATE` or `MKT_SHARE`, the names of the other columns of `alt_mapping`. Either is an error that names the column. Columns the model does not use may have any name.

## Examples

```
library(data.table)
set.seed(42)
N <- 100; J <- 3; beta_true <- c(1.0, -0.5)
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, V := drop(as.matrix(.SD) %*% beta_true), .SDcols = c("x1", "x2")]
dt[, prob := exp(V) / sum(exp(V)), by = id]
dt[, choice := as.integer(alt == sample(alt, 1, prob = prob)), by = id]

fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
summary(fit)
coef(fit)
AIC(fit)
predict(fit, type = "shares")
```

## Description

Estimates a multinomial probit model by Gibbs sampling with data augmentation (Albert & Chib 1993; McCulloch & Rossi 1994). The model is specified in utility differences against a base alternative: for choice situation  $i$  with  $J$  alternatives,  $w_i = X_i\beta + \epsilon_i$  with  $\epsilon_i \sim N_{J-1}(0, \Sigma)$ .

## Usage

```
run_mnprobit(
  data = NULL,
  id_col = NULL,
  alt_col = NULL,
  choice_col = NULL,
  covariate_cols = NULL,
  input_data = NULL,
  base_alt = NULL,
  use_asc = TRUE,
  prior = list(),
  mcmc = list(),
  keep_data = TRUE
)
```

## Arguments

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| data           | Data frame containing choice data (convenience workflow). Mutually exclusive with input_data.                                                                                                                                                                                                                                                                                                                                                                   |
| id_col         | Name of the column identifying choice situations (individuals).                                                                                                                                                                                                                                                                                                                                                                                                 |
| alt_col        | Name of the column identifying alternatives.                                                                                                                                                                                                                                                                                                                                                                                                                    |
| choice_col     | Name of the column indicating chosen alternative (1 = chosen, 0 = not chosen).                                                                                                                                                                                                                                                                                                                                                                                  |
| covariate_cols | Vector of names of columns to be used as covariates.                                                                                                                                                                                                                                                                                                                                                                                                            |
| input_data     | List output from <a href="#">prepare_mnp_data</a> (advanced workflow). Mutually exclusive with data.                                                                                                                                                                                                                                                                                                                                                            |
| base_alt       | Label of the base (reference) alternative used for utility differencing. If NULL (default), the first alternative in sort order is used.                                                                                                                                                                                                                                                                                                                        |
| use_asc        | Logical indicating whether to include alternative-specific constants (one intercept per non-base alternative in the differenced utilities), named ASC_<label>. No covariate may have one of these names; ASC dummies built by hand need use_asc = FALSE.                                                                                                                                                                                                        |
| prior          | Named list of prior settings, merged over defaults: <ul style="list-style-type: none"> <li>beta_bar Prior mean of <math>\beta</math> (default <math>\text{rep}(0, K)</math>).</li> <li>A Prior precision of <math>\beta</math> (default <math>0.01 * \text{diag}(K)</math>).</li> <li>nu Inverse-Wishart degrees of freedom (default <math>p + 3</math>).</li> <li>V Inverse-Wishart scale matrix (default <math>\text{nu} * \text{diag}(p)</math>).</li> </ul> |
| mcmc           | Named list of MCMC settings, merged over defaults: <ul style="list-style-type: none"> <li>R Total Gibbs iterations (default 10000).</li> </ul>                                                                                                                                                                                                                                                                                                                  |

burn Burn-in iterations discarded (default  $\text{floor}(R / 5)$ ).  
 thin Keep every thin-th post-burn-in draw (default 1).  
 seed Master RNG seed (default: drawn from R's RNG).  
 trace Print progress every trace iterations (default 0, silent).  
 keep\_data Logical. If TRUE (default), stores prepared data in the returned object.

## Details

Two workflows are supported:

**Convenience (default)** Supply data and column names. Data preparation (`prepare_mnp_data`) is handled automatically.

**Advanced** Call `prepare_mnp_data` yourself and pass the result via `input_data`.

**Identification.** The multinomial probit likelihood is invariant to a common rescaling  $(\beta, \Sigma) \rightarrow (c\beta, c^2\Sigma)$ . The sampler runs on the non-identified parameterization (unrestricted  $\Sigma$  with an inverse-Wishart prior) and identified quantities are computed by normalizing each kept draw by  $\sigma_{11}$ :  $\beta/\sqrt{\sigma_{11}}$  and  $\Sigma/\sigma_{11}$ . This is the McCulloch & Rossi (1994) default, which keeps all Gibbs conditionals conjugate and mixes better than the fully identified sampler of McCulloch, Polson & Rossi (2000). Reported coefficients, standard deviations, and credible intervals are posterior summaries of the identified draws.

**Reproducibility.** The sampler uses its own thread-safe RNG with one stream per (iteration, observation), so results are reproducible independent of the number of OpenMP threads (see `set_num_threads()`). When `mcmc$seed` is not supplied, a master seed is drawn from R's RNG, so `set.seed()` controls the run.

**Scope.** Balanced choice sets are required: every choice situation must contain the same  $J$  alternatives. To model an outside option, include it as explicit rows with zero covariates and set `base_alt` to its label.

## Value

A `choicer_mnp` object. S3 methods available: `summary()`, `coef()` (posterior means of identified coefficients), `vcov()` (posterior covariance of identified coefficient draws), `nobs()`. Posterior draws are stored in `$draws` (`beta / sigma` on the identified scale, `beta_raw / sigma_raw` unnormalized).

## Column names

No column the model uses (the `id`, `alternative`, `choice`, `covariate`, `weight`, `cluster`, `nest` or `decision-maker` column) may have a name starting with `.choicer_`, which `choicer` reserves for its working columns, and the `alternative` column may not be named `alt_int`, `N_OBS`, `N_CHOICES`, `TAKE_RATE` or `MKT_SHARE`, the names of the other columns of `alt_mapping`. Either is an error that names the column. Columns the model does not use may have any name.

## References

- Albert, J. H., & Chib, S. (1993). Bayesian Analysis of Binary and Polychotomous Response Data. *Journal of the American Statistical Association*, 88(422), 669-679.
- McCulloch, R., & Rossi, P. E. (1994). An exact likelihood analysis of the multinomial probit model. *Journal of Econometrics*, 64(1-2), 207-240.

## Examples

```
library(data.table)
set.seed(42)
N <- 200; J <- 3; beta_true <- c(1.0, -0.5)
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, U := drop(as.matrix(.SD) %*% beta_true) + rnorm(.N), .SDcols = c("x1", "x2")]
dt[, choice := as.integer(U == max(U)), by = id]

fit <- run_mnprobit(dt, "id", "alt", "choice", c("x1", "x2"),
                  mcmc = list(R = 500, burn = 100))
summary(fit)
coef(fit)
```

---

run\_mxlogit

*Runs mixed logit estimation*


---

## Description

Estimates a mixed logit model via simulated maximum likelihood.

## Usage

```
run_mxlogit(
  data = NULL,
  id_col = NULL,
  alt_col = NULL,
  choice_col = NULL,
  covariate_cols = NULL,
  random_var_cols = NULL,
  input_data = NULL,
  eta_draws = NULL,
  S = 100L,
  rc_dist = NULL,
  rc_mean = FALSE,
  rc_correlation = FALSE,
  use_asc = TRUE,
  theta_init = NULL,
  lower = NULL,
  upper = NULL,
  optimizer = NULL,
  control = list(),
  se_method = c("hessian", "bhhh", "sandwich", "cluster"),
  scale_vars = c("bhhh", "none", "sd", "mad", "iqr"),
  weights = NULL,
  outside_opt_label = NULL,
```

```

include_outside_option = FALSE,
draws = c("store", "generate"),
seed = NULL,
scramble = c("permuted", "none", "owen"),
keep_data = TRUE,
nloptr_opts = NULL,
weights_col = NULL,
cluster_col = NULL,
person_col = NULL,
polish = c("none", "newton", "bhhh"),
polish_control = list()
)

```

## Arguments

|                              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code>            | Data frame containing choice data (convenience workflow). Mutually exclusive with <code>input_data</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <code>id_col</code>          | Name of the column identifying choice situations.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <code>alt_col</code>         | Name of the column identifying alternatives.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <code>choice_col</code>      | Name of the column indicating chosen alternative (1/0).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <code>covariate_cols</code>  | Vector of column names for fixed covariates. None may take a name the model gives its parameters, or prints for them in <code>summary()</code> : $L_{<i><j>}$ and $\Sigma_{<i><j>}$ , with <code>use_asc = TRUE</code> $ASC_{<label>}$ , and with <code>rc_mean = TRUE</code> $\mu_{<variable>}$ and, for a log-normal coefficient, $\exp(\mu_{<variable>})$ .                                                                                                                                                                                                                                                                                                                                                                           |
| <code>random_var_cols</code> | Vector of column names for random coefficients.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <code>input_data</code>      | List output from <a href="#">prepare_mx1_data</a> (advanced workflow). Mutually exclusive with <code>data</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>eta_draws</code>       | Array of shape $K_w \times S \times U$ with standard normal draws, one $K_w \times S$ block per likelihood unit: $U$ is the number of decision makers when <code>input_data</code> was prepared with <code>person_col</code> ( <code>length(input_data\$Ti)</code> ), and the number of choice situations otherwise. Required for the advanced workflow; auto-generated from $S$ in the convenience workflow. Post-hoc methods ( <code>vcov(fit, type = )</code> , <a href="#">wesml_vcov</a> , <a href="#">conditional_tastes</a> , prediction) regenerate the Halton draws <a href="#">get_halton_normals</a> builds from the recorded draw count, so build <code>eta_draws</code> with it for them to reproduce the estimation draws. |
| <code>S</code>               | Integer number of Halton draws per decision maker, or per choice situation without <code>person_col</code> (convenience workflow only). Default 100.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>rc_dist</code>         | Integer vector indicating distribution of random coefficients (0 = normal, 1 = log-normal). Default: all normal.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <code>rc_mean</code>         | Logical indicating whether to estimate means for random coefficients.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <code>rc_correlation</code>  | Logical indicating whether random coefficients are correlated (convenience workflow). Ignored when <code>input_data</code> is used (taken from the prepared data).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>use_asc</code>         | Logical indicating whether to include alternative-specific constants: one per inside alternative with the outside option, otherwise one per alternative but the first                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

|              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <p>(outside_opt_label when it is given, else the first label in sort() order, which for a factor is the first of its levels present), whose constant is normalized to 0. With FALSE the utilities have no constants, so alternatives differ only through their covariates. An alternative's constant has no finite maximum-likelihood estimate when the alternative is never chosen in the data, or is chosen in every choice situation that offers it (the likelihood keeps rising as the constant falls, or grows); when the outside option or the base alternative is never chosen, or chosen wherever it is offered, the other constants have none together. The fit then warns (class choicer_unbounded_constants) before the optimizer runs, naming them; a finite bound in the direction a constant diverges (lower for a never-chosen alternative's, upper for an always-chosen one's) holds it, and that alternative is not named (the outside option's and the base's cases, which move every constant together, are named whatever the bounds).</p>                                                                                                                                                                                                                                  |
| theta_init   | <p>Initial parameter vector in natural-scale units. If NULL (default), the start is in the columns' units: zero for the <math>\beta</math>, <math>\mu</math> and ASC blocks and the Cholesky factor's off-diagonal entries; on its diagonal, <math>L_{pp} = 1/h_p</math> for a normal random coefficient, where <math>h_p</math> is the typical step of its column of <math>W</math>: the distance between its values for a column with two (a dummy: 1), the mean distance from a value that fills more than half of the column over the other rows (a mostly-zero column), and the standard deviation otherwise; a column with no variation starts at <math>L_{pp} = 1</math>. At the start, a step of <math>h_p</math> in the column then moves the utility by a random amount with standard deviation 1, whatever the column's units of measurement. A log-normal coefficient starts at <math>L_{pp} = 0.5</math>, the standard deviation of the logarithm of its random part, which has no units (that random part has median 1 in the column's units, so a log-normal covariate belongs in units where its coefficient is of order one). Where the default start falls outside lower/upper, it is moved to the nearest bound, with a message. A supplied theta_init is used as given.</p> |
| lower, upper | <p>Optional parameter bounds for the optimizer, in natural-scale units (mapped to the optimizer's coordinates when scale_vars != "none"); lower must not exceed upper, nor be Inf (nor upper -Inf). Each accepts three forms:</p> <p>NULL (default) Unbounded (-Inf/Inf).</p> <p><b>Unnamed numeric vector of length n_params</b> Full-length vector ordered exactly like theta_init (the nloptr-native form).</p> <p><b>Named numeric vector</b> Names must be a subset of the parameter names (<math>\beta</math> block: column names of X; <math>\mu</math> block: Mu_&lt;col&gt; (if rc_mean = TRUE); Cholesky block: L_&lt;i&gt;&lt;j&gt; for <math>i \geq j</math>; ASC block: ASC_&lt;level&gt;). Unlisted parameters default to <math>\pm\infty</math>. This is the recommended form for typical use, e.g. lower = c(L_11 = -5, L_22 = -5) to clip Cholesky diagonals.</p>                                                                                                                                                                                                                                                                                                                                                                                                              |
| optimizer    | <p>Optimizer to use: "nloptr" (default), "optim", or a custom function. See <a href="#">run_mnlogit</a> for details.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| control      | <p>List of optimizer-specific control parameters. With scale_vars other than "none", settings in parameter units apply to the scaled coordinates.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| se_method    | <p>Method for computing standard errors. One of "hessian" (default) for the analytical Hessian of the simulated log-likelihood, "bhhh" for the BHHH/outer-product-of-gradients (OPG) estimator, "sandwich" for the robust (Huber-White) variance <math>V = A^{-1}BA^{-1}</math> (bread <math>A</math> = weighted negated Hessian, meat <math>B</math> = weight-squared OPG), or "cluster" for the cluster-robust sandwich (requires</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

cluster\_col or a prepared input\_data with a cluster field). Use "sandwich" for valid inference under choice-based / WESML weighting, where the inverse-Hessian and ordinary BHHH are invalid; it reduces to the usual robust variance under uniform weights. BHHH needs no second derivatives and so costs less than the Hessian. For both, the work per decision maker involves only the parameters that enter that decision maker's likelihood (the fixed coefficients, the random coefficients' means and Cholesky factor, and the constants of the alternatives they faced), so it does not grow with the number of alternatives in the data; the parameters-by-parameters matrix and its inversion do. BHHH may underestimate standard errors in finite samples or away from the optimum. Any of these variances can also be recomputed post hoc via `vcov(fit, type = )`. Without `person_col`, clustering repairs the inference, not the likelihood: the cross-sectional likelihood still treats each choice situation as an independent draw from the mixing distribution, and on panel data is a less efficient composite likelihood for the same taste distribution. With `person_col` the likelihood unit is the decision maker, so scores are per decision maker and "sandwich" is already robust to dependence across a decision maker's choice situations; "cluster" is then needed only for coarser groups (e.g. households or markets) that nest decision makers.

#### scale\_vars

How the optimizer's coordinates are scaled. The choice does not change the estimator; it changes how fast, and whether, the optimizer reaches the maximum. "bhhh" (default) scales each parameter by the inverse square root of the diagonal of the BHHH (outer product of gradients) matrix at the start values, rounded to a power of two, from one extra gradient pass, so that the BHHH diagonal, an estimate of the curvature, is near one at the start values in every coordinate: it reaches the constants and the Cholesky factor, whose curvatures can differ by orders of magnitude that no column scale reaches (the constants of large and small alternatives, say). A start near the estimates (`theta_init`) gives it a better diagonal. Where the diagonal is unusable (no finite, positive entry, as when the likelihood overflows at the start values), the parameters are left unscaled, with a message. "none" hands the optimizer the parameters as they are (the default in choicer 0.2.x). With "sd" (sample standard deviation), "mad" (`stats::mad`, i.e.  $1.4826 \times$  median absolute deviation; SD-equivalent under normality) or "iqr" (`stats::IQR(x) / 1.349`; also SD-equivalent under normality), the optimizer works on the parameters the design would have with every column of  $X$  and  $W$  divided by the chosen scale; the constants keep their scale, the data are not divided, and the fit holds no scaled copy of them. Robust scales ("mad"/"iqr") better capture the bulk for heavy-tailed columns where SD is dominated by outliers, but `stats::mad` can return zero when more than half of a column's entries are identical (e.g., a sparse 0/1 dummy) and will then trigger the same near-constant-column error as "sd". The parameters of log-normal random coefficients (`rc_dist == 1`) keep their scale under these three, since the shifted log-normal parameterization does not admit a closed-form rescaling under multiplicative column scaling. With any choice but "none" (so by default), the optimizer, a custom one included, works in the scaled coordinates (`theta_init`, lower and upper are given in natural units and mapped), so settings in parameter units in control (`nloptr's xtol_abs`; `optim's parscale`, and `pgtol`, which then applies to the scaled gradient) apply to the scaled coordi-

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                        | <p>nates; "none" keeps the parameters' units. Coefficients are reported in natural units and the standard errors computed in them, from the information matrix equilibrated before it is inverted, whatever the choice, so, where the likelihood is well identified and every choice reaches its maximum, reported quantities do not depend on this choice beyond the optimizer's tolerance.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| weights                | <p>Optional weight vector (convenience workflow), one weight per choice situation in ascending-id order (see <a href="#">prepare_mxl_data</a>). With <code>person_col</code>, situations are reordered by decision maker, so a positional vector is accepted only when that leaves the id order unchanged; otherwise supply <code>weights_col</code>. If NULL, equal weights are used. All weights must be finite and strictly positive. With <code>person_col</code> they are decision-maker weights, constant within each decision maker: the objective is <math>\sum_n w_n \log L_n</math>.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| outside_opt_label      | <p>(Convenience workflow.) The label of the outside option or of the base alternative, given in the class of the alternative column. With <code>include_outside_option = TRUE</code> it names the outside option's row of <code>alt_mapping</code>. Without the outside option it names the base alternative, coded first (with constants, its constant is normalized to 0); it must be one of the alternatives, and NULL (the default) makes the first alternative in sort order the base.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| include_outside_option | <p>Logical whether to include an outside option (convenience workflow).</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| draws                  | <p>Draw storage mode. One of "store" (default) or "generate". "store" pre-materializes the full <math>K_w \times S \times U</math> Halton cube (<math>8K_wSU</math> bytes), one block per likelihood unit (U decision makers with <code>person_col</code>, choice situations otherwise; existing behavior, exact reproducibility). It supports at most <math>2^{31} - 1</math> points (<math>S \times U</math>). <code>predict()</code>, <code>logsum()</code>, <code>consumer_surplus()</code>, <code>elasticities()</code>, <code>diversion_ratios()</code>, <code>blp()</code> and <code>gof()</code> (hence <code>summary()</code>'s fit statistics) integrate over one block per choice situation instead (see <a href="#">get_halton_normals</a>). With up to 128 random coefficients they form each block on the fly, in parallel, from the points <code>get_halton_normals(S, N, K_w)</code> would hold for the <math>N</math> choice situations, mapped by R's <code>qnorm()</code>, so they never hold that cube and have no limit on the number of points; with more, they build the cube. Their draws, and so their results, are the cube's, bit for bit where <code>randtoolbox</code> and <code>choicer</code> are compiled with the same floating-point contraction, as on CRAN's platforms (otherwise some of the underlying uniforms differ in their last bit, which moves those draws by at most a few parts in <math>10^9</math>, in the far tails); sums over choice situations are bit for bit at one thread, and at more threads vary in their last bits with the threads' order, as they always did. <code>blp()</code> may keep the draws it forms across its iterations, within its <code>keep_draws_bytes</code> (4 GiB by default), with either storage mode. In the convenience workflow, <code>run_mxlogit()</code> warns before it builds an estimation cube above 1 GiB: the cube is held while the model is fitted, and <code>vcov(type = )</code>, <code>wesml_vcov</code> and <code>conditional_tastes</code> rebuild it whole. "generate" computes each unit's draws on-the-fly in C++ from a stored seed, eliminating the <math>O(U)</math> cube; recommended for memory-constrained or large-N settings. With <code>scramble = "permuted"</code>, each base-<math>b</math> digit position in each dimension receives a seeded permutation shared across sequence indices. This is not Owen's nested-uniform scramble and does not carry standard randomized-QMC unbiasedness or error-estimation guarantees. Only supported in the convenience workflow.</p> |

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| seed           | Integer master seed for the on-the-fly generator. Used only when draws = "generate". If NULL (default), a seed is drawn from R's RNG at call time (so <code>set.seed()</code> governs reproducibility). Ignored when draws = "store".                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| scramble       | Scrambling mode for on-the-fly Halton draws. One of "permuted" (default) for seeded position-wise digit permutations or "none" for plain Halton (identity permutations). The historical value "owen" is accepted with a deprecation warning as an alias for "permuted"; the implementation is not Owen's nested-uniform scramble. "none" uses the points of the <code>randtoolbox</code> sequence, mapped to normals by choicer's own inverse normal CDF, which is within about $10^{-14}$ of <code>qnorm()</code> : its draws agree with store mode's to that precision, not bit for bit. Simulation-draw sensitivity should be assessed by increasing S and, for "permuted", varying seed. Used only when draws = "generate".                                                                                                                                                                                                                                                                                                                                          |
| keep_data      | Logical. If TRUE (default), stores prepared data in the returned object for post-estimation functions.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| nloptr_opts    | Deprecated. Use <code>optimizer</code> and <code>control</code> instead.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| weights_col    | Optional name of a column in data holding a per-row weight (constant within each choice situation, finite and strictly positive). Mutually exclusive with <code>weights</code> ; the recommended way to pass WESML weights from <code>sample_by_choice</code> / <code>wesml_weights</code> , since alignment is by id rather than by position. Convenience workflow only. If data carries choice-based-sampling provenance (a "choice_sampling" attribute, as attached by <code>sample_by_choice</code> / <code>wesml_weights</code> ) and neither <code>weights</code> nor <code>weights_col</code> is supplied, the recorded weight column is auto-detected and applied (with a message); if that column is absent the call errors rather than silently fitting an unweighted model under a WESML label. With <code>person_col</code> the weight must also be constant within each decision maker, and WESML is not supported: choice-based weights vary with the chosen alternative across a decision maker's situations, so they cannot weight the panel likelihood. |
| cluster_col    | Optional name of a column in data holding cluster labels for cluster-robust standard errors (e.g. a person id when the same decision maker contributes several choice situations to a cross-sectional fit). Must be constant within each <code>id_col</code> , and within each decision maker when <code>person_col</code> is used (clusters must nest decision makers). Supplying <code>cluster_col</code> without an explicit <code>se_method</code> selects <code>se_method = "cluster"</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| person_col     | Optional name of the column in data identifying decision makers (respondents). When supplied, all choice situations of a decision maker share one draw of the random coefficients (panel likelihood); <code>id_col</code> must still identify choice situations uniquely across the data set. NULL (default) makes each choice situation its own decision maker (cross-sectional likelihood). Convenience workflow only; in the advanced workflow pass <code>person_col</code> to <code>prepare_mx1_data</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| polish         | A Newton phase after the optimizer (see the section below): "none" (default) has none; "newton" steps on the analytic Hessian of the simulated log-likelihood, "bhhh" on the BHHH (outer product of the gradients) matrix, both with the fit's draws and weights.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| polish_control | List of settings of the Newton phase, any of <code>max_iter</code> (20; 0 evaluates the decrement at the optimizer's solution without stepping), <code>tol_decrement</code> ( $10^{-8}$ , in log-likelihood units) and <code>tol_step</code> ( $10^{-12}$ ; a step shorter than this, relative                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

to the parameters' magnitudes plus their scales in the optimizer's coordinates, stops the polish unconverged).

## Details

Two workflows are supported:

**Convenience** Supply data and column names. Data preparation ([prepare\\_mxl\\_data](#)) and Halton draw generation ([get\\_halton\\_normals](#)) are handled automatically.

**Advanced** Call [prepare\\_mxl\\_data](#) and [get\\_halton\\_normals](#) yourself, then pass the results via `input_data` and `eta_draws`.

**Cross-section vs panel.** With `person_col`, each decision maker  $n$  draws one taste vector  $\beta_n$  from the mixing distribution  $f(\beta \mid \theta)$  and keeps it in all of their choice situations  $t = 1, \dots, T_n$ . The likelihood integrates the product of their conditional logit probabilities (Revelt and Train 1998),

$$L_n(\theta) = \int \prod_{t=1}^{T_n} P_{nt}(j_{nt} \mid \beta) f(\beta \mid \theta) d\beta,$$

simulated with  $S$  draws per decision maker, so repeated choices speak to each decision maker's tastes, and [conditional\\_tastes](#) recovers what they reveal. Without `person_col` (the default), each choice situation is an independent draw from the mixing distribution:  $L_i(\theta) = \int P_i(j_i \mid \beta) f(\beta \mid \theta) d\beta$ . That is the right model when tastes are redrawn in every situation (or each decision maker chooses once). On panel data with stable tastes it is a composite (pseudo-) likelihood: it still targets the population mixing distribution, but it ignores the within-person correlation of choices, so it is less efficient, its standard errors must be clustered by decision maker (`cluster_col`), and its log-likelihood, AIC and BIC are not comparable with those of the panel fit.

## Value

A `choicer_mxl` object (inherits from `choicer_fit`). Standard S3 methods available: `summary()`, `coef()`, `vcov()`, `logLik()`, `AIC()`, `BIC()`, `nobs()`. A panel fit (`person_col`) also carries `n_persons`, the number of decision makers; `nobs()` remains the number of choice situations, which is also the  $n$  in BIC's  $\log n$  penalty. The variance is computed once the optimizer has stopped. If that step fails with an error (the kernels' memory check, see [set\\_num\\_threads](#), say), the fit returns its estimates without it (`vcov` NULL, NA standard errors, the error's message in `variance_error`) and warns, naming the `vcov(fit, type = )` that computes the variance later from a fit that keeps its data (`keep_data = TRUE`); the memory check's message names the threads that fit or how to lift its limit. With `polish`, `fit$optimizer$polish` records the Newton phase: `method`, `converged`, `iterations` (steps taken) and `damped`, `decrement` (NA where the curvature on the parameters moved was not positive definite) and `lambda` (its damping) at the last curvature evaluated, `message` (why it stopped), `evaluations` (of the objective and the curvature), `active`, `held` and `flat` (the parameters on a bound, held, and left out of the last steps for want of usable curvature), `improvement` (the log-likelihood gained), `error`, `elapsed_time` (the polish's seconds, which `fit$optimizer$elapsed_time` includes), and `optimizer`, the optimizer's own convergence, `message`, `iterations` and `elapsed_time`. A converged polish sets convergence to the optimizer's success code (1 for `nloptr`, 0 for `optim` and a custom optimizer) and leads `message` with its verdict; otherwise both keep the optimizer's, the polish's reason appended to the message. A polish that stops with an error (the Hessian's memory check, say) keeps its last accepted point and warns (class `choicer_polish_failed`).

### Newton phase (polish)

The polish does not change the estimator. From wherever the optimizer stopped (at its tolerance or at maxeval), it finishes the climb to the maximum the optimizer approached, which first-order methods such as L-BFGS reach slowly when curvatures differ by orders of magnitude (thousands of constants, say), and certifies it. It converges when the Newton decrement  $\nu = \frac{1}{2}g^\top C^{-1}g$ , with  $g$  the gradient and  $C$  the curvature on the parameters it moves, falls below tol\_decrement. On a positive definite Hessian,  $\sqrt{2\nu}$  is the distance from the estimates to the maximum of the log-likelihood's quadratic model in Hessian standard errors: at the default  $10^{-8}$ , every linear combination of the estimates lies within  $1.4 \times 10^{-4}$  of its standard error of that maximum. BHHH is positive semi-definite everywhere and cheaper per step, but converges linearly; where the Hessian  $H$  is positive definite, its  $\sqrt{2\nu}$  is that distance in the standard errors of the sandwich  $H^{-1}BH^{-1}$  (the robust standard errors of an unweighted fit), but on its own it shows a stationary point, not a maximum. "bhhh" needs at least as many likelihood units as the parameters it moves.

Where the curvature is not positive definite the steps are damped (Levenberg-Marquardt); they respect lower/upper and pass a line search. The constants of inside alternatives without a finite estimate (see use\_asc) are held at the optimizer's values, and the certificate concerns the other parameters given them. Where the base alternative or the outside option is the one without, the other constants diverge together, and the polish does not report convergence, unless the reference is never chosen and the constant of an alternative that is chosen is held or bounded above, which stops them. A random coefficient whose variance's maximum is at zero drives its log-Cholesky diagonal toward  $-\infty$  by Newton steps of  $\frac{1}{2}$  to 1, each shrinking the decrement by a factor of about  $e$ : "newton" may certify a point where the variance is negligible but that diagonal, and its standard error, depend on tol\_decrement; "bhhh", whose curvature in that direction vanishes faster, comes to hold the diagonal and stops unconverged. A parameter without usable curvature (no information at the point, or a curvature that has all but vanished, giving a step more than 100 times its scale in the optimizer's coordinates) is held for the step and stops the polish unconverged. Under scale\_vars = "none" those scales are the parameters' natural units, so a badly scaled column far from its maximum can be held as if its curvature had vanished; the default "bhhh" scaling avoids this. The standard errors are computed at the polished estimates, by se\_method.

Each step costs a curvature evaluation and a line search. A smaller maxeval (nloptr) or maxit (optim) in control hands the polish a point sooner, which saves time where the optimizer's last iterations creep; from farther away the steps are shortened (and damped where the curvature is not positive definite), and more of them are needed.

### Column names

No column the model uses (the id, alternative, choice, covariate, weight, cluster, nest or decision-maker column) may have a name starting with .choicer\_, which choicer reserves for its working columns, and the alternative column may not be named alt\_int, N\_OBS, N\_CHOICES, TAKE\_RATE or MKT\_SHARE, the names of the other columns of alt\_mapping. Either is an error that names the column. Columns the model does not use may have any name.

### References

Revelt, D. and Train, K. (1998). Mixed logit with repeated choices: households' choices of appliance efficiency level. *Review of Economics and Statistics* 80(4), 647-657.

**See Also**

[conditional\\_tastes](#), [run\\_hmnlogit](#) (the hierarchical Bayes counterpart)

**Examples**

```
library(data.table)
set.seed(42)
N <- 100; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N), w2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]

fit <- run_mxlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = "x1", random_var_cols = c("w1", "w2"), S = 50L
)
summary(fit)
```

---

run\_nestlogit

*Runs nested logit estimation*


---

**Description**

Estimates a nested logit model via maximum likelihood.

**Usage**

```
run_nestlogit(
  data = NULL,
  id_col = NULL,
  alt_col = NULL,
  choice_col = NULL,
  covariate_cols = NULL,
  nest_col = NULL,
  input_data = NULL,
  use_asc = TRUE,
  theta_init = NULL,
  param_names = NULL,
  optimizer = NULL,
  control = list(),
  weights = NULL,
  weights_col = NULL,
  outside_opt_label = NULL,
  include_outside_option = FALSE,
  keep_data = TRUE,
  se_method = c("hessian", "numeric", "bhhh", "sandwich", "cluster"),
```

```

cluster_col = NULL,
nloptr_opts = NULL,
scale_vars = c("none", "sd", "mad", "iqr", "bhhh")
)

```

## Arguments

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| data           | Data frame containing choice data (convenience workflow). Mutually exclusive with input_data.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| id_col         | Name of the column identifying choice situations.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| alt_col        | Name of the column identifying alternatives.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| choice_col     | Name of the column indicating chosen alternative (1/0).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| covariate_cols | Vector of column names for covariates. None may be named like a generated parameter: Lambda_<k>, or ASC_<label> with use_asc = TRUE.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| nest_col       | Name of the column mapping each alternative to its nest (convenience workflow).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| input_data     | List containing prepared input data for estimation (advanced workflow). Mutually exclusive with data.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| use_asc        | Logical indicating whether to include alternative specific constants (ASCs). An alternative's constant has no finite maximum-likelihood estimate when the alternative is chosen in every choice situation that offers it, or is never chosen in the data (the likelihood keeps rising as the constant grows, or falls); when the outside option or the base alternative is chosen wherever it is offered, or never chosen, the other constants have none together. For a never-chosen alternative, the base included, this holds while the parameter of its nest is at most 1. The fit then warns (class choicer_unbounded_constants) before the optimizer runs, naming them. |
| theta_init     | Optional initial parameter vector in natural units, ordered as the coefficients, the nest parameters (Lambda_<k>, one per nest of two or more alternatives) and the constants. If NULL: 0 for the coefficients and constants, 0.5 for each nest parameter.                                                                                                                                                                                                                                                                                                                                                                                                                    |
| param_names    | Optional vector of parameter names, which must be unique. If NULL, default names are generated.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| optimizer      | Optimizer to use: "nloptr" (default), "optim", or a custom function. See <a href="#">run_mnlogit</a> for details.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| control        | List of optimizer-specific control parameters. With scale_vars other than "none", settings in parameter units apply to the scaled coordinates.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| weights        | Optional weight vector (convenience workflow). If NULL, equal weights are used. All weights must be finite and strictly positive.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| weights_col    | Optional name of a column in data holding per-row weights (convenience workflow only). The column must be constant within each id_col (one weight per choice situation) and is collapsed accordingly. Mutually exclusive with weights. All weights must be finite and strictly positive. Used for choice-based / WESML weighting; pair with se_method = "sandwich" for valid inference.                                                                                                                                                                                                                                                                                       |

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| outside_opt_label      | (Convenience workflow.) The label of the outside option or of the base alternative, given in the class of the alternative column. With <code>include_outside_option = TRUE</code> it labels the outside option, which the model adds: rows of the data that carry it are dropped, and it names the outside option's row of <code>alt_mapping</code> . Without the outside option it names the base alternative, coded first (with constants, its constant is normalized to 0); it must be one of the alternatives, and <code>NULL</code> (the default) makes the first alternative in sort order the base.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| include_outside_option | Logical whether to include an outside option (convenience workflow).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| keep_data              | Logical. If <code>TRUE</code> (default), stores prepared data in the returned object for post-estimation functions.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| se_method              | Method for computing standard errors: "hessian" (default, analytical Hessian via <code>nl_loglik_hessian_parallel</code> ), "numeric" (finite-difference oracle via <code>nl_loglik_numeric_hessian</code> ), "bhhh" (outer product of gradients via <code>nl_bhhh_parallel</code> ), "sandwich" (robust Huber-White / WESML variance $A^{-1}BA^{-1}$ ), or "cluster" (cluster-robust sandwich; requires <code>cluster_col</code> or a prepared <code>input_data</code> with a <code>cluster</code> field). Use "sandwich" under choice-based / WESML weighting. Any of these can also be recomputed post hoc via <code>vcov(fit, type = )</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| cluster_col            | Optional name of a column in data holding cluster labels for cluster-robust standard errors (e.g. a person id when the same decision maker contributes several choice situations). Must be constant within each <code>id_col</code> . Supplying <code>cluster_col</code> without an explicit <code>se_method</code> selects <code>se_method = "cluster"</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| nloptr_opts            | Deprecated. Use <code>optimizer</code> and <code>control</code> instead.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| scale_vars             | How the optimizer's coordinates are scaled. The choice does not change the estimator; it changes how fast, and whether, the optimizer reaches the maximum. One of "none" (default), "sd" (sample standard deviation), "mad" ( <code>stats::mad</code> ), "iqr" ( <code>stats::IQR(x) / 1.349</code> ), or "bhhh". With "sd", "mad" or "iqr", the optimizer works on the coefficients $X$ would have with every column divided by the chosen scale (each coefficient times its column's scale), to improve the conditioning of its problem; the nest parameters and constants keep their scale, and the data are not divided. "bhhh" scales every parameter, nest parameters included, by the inverse square root of the diagonal of the BHHH (outer product of gradients) matrix at the start values, rounded to a power of two, from one extra gradient pass. Its scales are fixed at the start values, so when the nest parameters end far from their start (0.5) they can mislead the optimizer, which then converges more slowly than without them; hence "none" is the default here. Where the diagonal is unusable (no finite, positive entry), the parameters are left unscaled, with a message. With any choice but "none", the optimizer, a custom one included, works in the scaled coordinates ( <code>theta_init</code> , in natural units, and the nest parameters' lower bound of $1e-16$ are mapped), so settings in parameter units in <code>control</code> ( <code>nloptr</code> 's <code>xtol_abs</code> ; <code>optim</code> 's <code>parscale</code> , and <code>pgtol</code> , which then applies to the scaled gradient) apply to the scaled coordinates; "none" keeps the parameters' units. Coefficients are reported in natural units and the standard errors computed in them, from the information matrix equilibrated before it is inverted, whatever the choice, so, where the likelihood is well identified and every choice reaches its maximum, reported quantities do not depend on this choice beyond the optimizer's tolerance. |

## Details

Two workflows are supported:

**Convenience** Supply data and column names (including `nest_col`). Data preparation ([prepare\\_nl\\_data](#)) is handled automatically.

**Advanced** Call [prepare\\_nl\\_data](#) (or build the input list manually) and pass it via `input_data`.

## Value

A `choicer_nl` object (inherits from `choicer_fit`). Standard S3 methods available: `summary()`, `coef()`, `vcov()`, `logLik()`, `AIC()`, `BIC()`, `nobs()`. The variance is computed once the optimizer has stopped. If that step fails with an error (an allocation in R, say), the fit returns its estimates without it (`vcov` NULL, NA standard errors, the error's message in `variance_error`) and warns, naming the `vcov(fit, type = )` that computes the variance later from a fit that keeps its data (`keep_data = TRUE`). An allocation that fails inside the compiled Hessian or BHHH loops still terminates R.

## Column names

No column the model uses (the `id`, `alternative`, `choice`, `covariate`, `weight`, `cluster`, `nest` or `decision-maker` column) may have a name starting with `.choicer_`, which `choicer` reserves for its working columns, and the `alternative` column may not be named `alt_int`, `N_OBS`, `N_CHOICES`, `TAKE_RATE` or `MKT_SHARE`, the names of the other columns of `alt_mapping`. Either is an error that names the column. Columns the model does not use may have any name.

## Examples

```
library(data.table)
set.seed(42)
N <- 100; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, nest := ifelse(alt <= 2, "A", "B")]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]

fit <- run_nestlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = c("x1", "x2"), nest_col = "nest"
)
summary(fit)
```

## Description

Subsamples whole choice situations from a population data set according to fixed per-stratum quotas, where strata are defined by the *chosen* alternative. The input data set is treated as the population, so the population shares  $Q(j)$  are known exactly; the returned sample carries a ready-to-use WESML weight column (see [wesml\\_weights](#)).

## Usage

```
sample_by_choice(
  data,
  id_col,
  alt_col,
  choice_col,
  n_per_alt = NULL,
  frac_per_alt = NULL,
  seed = NULL,
  weight_name = ".wesml_weight",
  outside_opt_label = NULL,
  include_outside_option = FALSE
)
```

## Arguments

|                                                        |                                                                                                                                                                                                                                |
|--------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data, id_col, alt_col, choice_col</code>         | As in <a href="#">wesml_weights</a> .                                                                                                                                                                                          |
| <code>n_per_alt</code>                                 | Either a single integer applied to every stratum, or a named integer vector of per-stratum counts (names matched to <code>as.character(alt)</code> , covering all strata). Mutually exclusive with <code>frac_per_alt</code> . |
| <code>frac_per_alt</code>                              | Either a single fraction in $[0, 1]$ applied to every stratum, or a named numeric vector of per-stratum fractions. Mutually exclusive with <code>n_per_alt</code> .                                                            |
| <code>seed</code>                                      | Optional integer seed for reproducible sampling.                                                                                                                                                                               |
| <code>weight_name</code>                               | Name of the attached weight column (default <code>".wesml_weight"</code> ).                                                                                                                                                    |
| <code>outside_opt_label, include_outside_option</code> | As in <a href="#">wesml_weights</a> (for an implicit outside good).                                                                                                                                                            |

## Details

Sampling is by choice situation (`id`), never by row: all alternative-rows of a sampled situation are kept together. Sampling is **without replacement**.

## Value

A data.table subsample with the weight column appended and `"Q"`, `"H"`, and `"choice_sampling"` attributes (the last records the scheme, shares, quotas, and `meat = "robust"`).

## References

Manski, C. F. and Lerman, S. R. (1977). *Econometrica* 45(8), 1977-1988.

**See Also**

[wesml\\_weights](#), [run\\_mxlogit](#)

**Examples**

```
library(data.table)
set.seed(1)
N <- 600L; J <- 3L
pop <- data.table(id = rep(seq_len(N), each = J), alt = rep(1:J, N))
pop[, x1 := rnorm(.N)]
pop[, w1 := rnorm(.N)]
pop[, choice := as.integer(seq_len(.N) == sample.int(.N, 1L)), by = id]

s <- sample_by_choice(pop, "id", "alt", "choice", n_per_alt = 50L, seed = 1L)
attr(s, "choice_sampling")$H # realized sample shares
head(s[["wesml_weight"]])
```

---

set\_num\_threads

*Set the number of OpenMP threads used by choicer*


---

**Description**

Set the number of OpenMP threads used by choicer

**Usage**

```
set_num_threads(n_threads)
```

**Arguments**

n\_threads      Positive integer number of threads.

**Details**

The setting applies to every choicer kernel run afterwards in the session (it calls `omp_set_num_threads()`; [thread\\_info\(\)](#) reports it. Each thread of the mixed logit's compiled routines (behind the log-likelihood and its gradient, the Hessian, the BHHH matrix, the robust and clustered meats and the conditional tastes) needs its own working memory, besides what they share. Before allocating it, they compare an upper estimate of the call's total with the machine's total physical memory and, when it would not fit, stop with an error that gives the number of threads that fits; they do not lower the thread count to make it fit. Memory already in use (the data, store-mode draws, other processes) is not counted, nor is the inversion in R of the resulting matrices, so passing the check does not promise that the call fits; where the physical memory cannot be read and the option is unset, there is no check. `options(choicer.max_memory = <bytes>)` sets the limit instead, for a container or a cluster job with less memory than the machine, and `Inf` lifts the check.

**Value**

Invisibly returns `NULL`.

---

|                    |                                                     |
|--------------------|-----------------------------------------------------|
| simulate_hmnl_data | <i>Simulate hierarchical multinomial logit data</i> |
|--------------------|-----------------------------------------------------|

---

### Description

Generates synthetic panel choice data from the hierarchical (random coefficients + alternative-level random effects) logit DGP: respondents  $i = 1 \dots N$  face  $T$  choice situations each, with utilities

$$U_{ijt} = x'_{ijt}\gamma_i + \delta_j + \epsilon_{ijt}, \quad U_{iot} = \epsilon_{iot},$$

i.i.d. Gumbel shocks (including a shock on the outside option, whose systematic utility is 0),  $\beta_i \sim N(\beta, W)$  with  $\gamma_{ik} = \beta_{ik}$  or  $\exp(\beta_{ik})$  per `rc_dist`, and  $\delta_j = z'_j\theta + \xi_j$  with  $\xi_j \sim N(0, \sigma_d^2)$ . Covariates are Uniform(-1, 1); the alternative-level covariates  $z^*$  are constant within each alternative.

### Usage

```
simulate_hmnl_data(
  N = 500,
  T = 10,
  J = 4,
  beta = c(0.8, -0.6),
  W = NULL,
  theta = c(0.5, -0.4),
  sigma_d = 0.5,
  Z = NULL,
  rc_dist = NULL,
  include_outside = TRUE,
  seed = 123,
  vary_choice_set = FALSE
)
```

### Arguments

|         |                                                                                                                                                                 |
|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| N       | Number of respondents.                                                                                                                                          |
| T       | Number of choice situations per respondent.                                                                                                                     |
| J       | Number of inside alternatives.                                                                                                                                  |
| beta    | Population means of the structural random coefficients (length $K_x = \text{length}(\text{beta})$ ; chain scale for log-normal coordinates).                    |
| W       | Covariance of the random coefficients ( $K_x \times K_x$ ). Defaults to <code>diag(0.5, K_x)</code> .                                                           |
| theta   | Mean-function coefficients for $\delta_j = z'_j\theta + \xi_j$ ; the first entry is the intercept, entries 2..P load on the alternative-level covariates.       |
| sigma_d | Standard deviation of the alternative-level effects $\xi_j$ .                                                                                                   |
| Z       | Optional $J \times (\text{length}(\text{theta}) - 1)$ matrix of alternative-level covariates (excluding the intercept). Default NULL draws them Uniform(-1, 1). |

|                 |                                                                                                                                                                                                                                                                                                                                                                 |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| rc_dist         | Integer vector (length K_x): 0L for normal, 1L for log-normal coordinates. Default NULL is all-normal.                                                                                                                                                                                                                                                          |
| include_outside | Logical; if TRUE (default) every choice set also contains an outside option with systematic utility 0 and its own Gumbel shock. The outside good is <i>implicit</i> in the returned data (matching the <code>prepare_hmnl_data()</code> convention): no physical row is emitted, and a choice situation the outside option wins has an all-zeros choice column. |
| seed            | Random seed (NULL skips <code>set.seed()</code> ).                                                                                                                                                                                                                                                                                                              |
| vary_choice_set | Logical; if TRUE choice-set size is sampled uniformly from 2:J per task. Default FALSE.                                                                                                                                                                                                                                                                         |

### Details

Log-normal coordinates are reported on the *chain* (log) scale in `true_params$beta` — the scale on which the estimator’s hierarchy operates — while entering utility as `exp(beta_ik)`.

### Value

A `choicer_sim` object. `true_params` contains `beta`, `W`, `theta`, `sigma_d`, the realized `delta` and `xi` vectors, the full mean-function design `Z` (intercept first), and `rc_dist`.

### Examples

```
sim <- simulate_hmnl_data(N = 100, T = 4, J = 4, seed = 123)
print(sim)
sim>true_params$delta
```

---

|                    |                                                      |
|--------------------|------------------------------------------------------|
| simulate_hmnp_data | <i>Simulate hierarchical multinomial probit data</i> |
|--------------------|------------------------------------------------------|

---

### Description

Generates synthetic panel choice data from the hierarchical probit DGP with iid normal utility shocks:

$$U_{ijt} = x'_{ijt}\beta_i + \delta_j + \epsilon_{ijt}, \quad U_{iot} = \epsilon_{iot}, \quad \epsilon \sim N(0, \sigma^2),$$

choice by `argmax` within the task. The outside option is stochastic — it carries its own  $N(0, \sigma^2)$  shock on top of systematic utility 0, exactly as in the estimator.  $\beta_i \sim N(\beta, W)$  (normal only) and  $\delta_j = z'_j\theta + \xi_j$ ,  $\xi_j \sim N(0, \sigma_d^2)$ , as in `simulate_hmnl_data()`.

**Usage**

```
simulate_hmnp_data(
  N = 500,
  T = 10,
  J = 4,
  beta = c(0.8, -0.6),
  W = NULL,
  theta = c(0.5, -0.4),
  sigma_d = 0.5,
  Z = NULL,
  include_outside = TRUE,
  seed = 123,
  vary_choice_set = FALSE,
  sigma = 1
)
```

**Arguments**

|                 |                                                                                                                                                                                                                                                                                                                                                                 |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| N               | Number of respondents.                                                                                                                                                                                                                                                                                                                                          |
| T               | Number of choice situations per respondent.                                                                                                                                                                                                                                                                                                                     |
| J               | Number of inside alternatives.                                                                                                                                                                                                                                                                                                                                  |
| beta            | Population means of the structural random coefficients (length $K_x = \text{length}(\text{beta})$ ); all coordinates are normal.                                                                                                                                                                                                                                |
| W               | Covariance of the random coefficients ( $K_x \times K_x$ ). Defaults to $\text{diag}(0.5, K_x)$ .                                                                                                                                                                                                                                                               |
| theta           | Mean-function coefficients for $\delta_j = z_j' \theta + \xi_j$ ; the first entry is the intercept, entries 2..P load on the alternative-level covariates.                                                                                                                                                                                                      |
| sigma_d         | Standard deviation of the alternative-level effects $\xi_j$ .                                                                                                                                                                                                                                                                                                   |
| Z               | Optional $J \times (\text{length}(\text{theta}) - 1)$ matrix of alternative-level covariates (excluding the intercept). Default NULL draws them Uniform(-1, 1).                                                                                                                                                                                                 |
| include_outside | Logical; if TRUE (default) every choice set also contains an outside option with systematic utility 0 and its own normal shock. The outside good is <i>implicit</i> in the returned data (matching the <code>prepare_hmnp_data()</code> convention): no physical row is emitted, and a choice situation the outside option wins has an all-zeros choice column. |
| seed            | Random seed (NULL skips <code>set.seed()</code> ).                                                                                                                                                                                                                                                                                                              |
| vary_choice_set | Logical; if TRUE choice-set size is sampled uniformly from 2:J per task. Default FALSE.                                                                                                                                                                                                                                                                         |
| sigma           | Standard deviation of the iid utility shocks (DGP scale).                                                                                                                                                                                                                                                                                                       |

**Details**

The iid-probit likelihood identifies parameters only up to the common scale  $\sigma$ , so `true_params` is reported on the *identified* scale:  $\text{beta} = \beta/\sigma$ ,  $W = W/\sigma^2$ ,  $\text{theta} = \theta/\sigma$ ,  $\text{sigma}_d = \sigma_d/\sigma$ ,  $\text{delta} = \delta/\sigma$ ,  $\text{xi} = \xi/\sigma$ . With the default `sigma = 1` the DGP scale and the identified scale coincide.

**Value**

A choicer\_sim object. true\_params contains beta, W, theta, sigma\_d, the realized delta and xi, and the full mean-function design Z — all on the identified scale (see Details).

**Examples**

```
sim <- simulate_hmnp_data(N = 100, T = 4, J = 4, seed = 123)
print(sim)
sim$true_params$delta
```

---

|                   |                                        |
|-------------------|----------------------------------------|
| simulate_mnl_data | <i>Simulate multinomial logit data</i> |
|-------------------|----------------------------------------|

---

**Description**

Generates synthetic choice data with i.i.d. Gumbel errors, optionally with varying choice-set sizes and an outside option (alt = 0). Choices are determined by argmax of utility; covariates are drawn as Uniform(-1, 1).

**Usage**

```
simulate_mnl_data(
  N = 5000,
  J = 5,
  beta = c(0.8, -0.6),
  delta = NULL,
  seed = 123,
  outside_option = TRUE,
  vary_choice_set = TRUE
)
```

**Arguments**

|                 |                                                                                                                                      |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------|
| N               | Number of choice situations.                                                                                                         |
| J               | Number of inside alternatives.                                                                                                       |
| beta            | Fixed coefficients for $x_1 \dots x_{K_x}$ (length $K_x = \text{length}(\text{beta})$ ).                                             |
| delta           | Alternative-specific constants for inside alternatives (length J). Defaults to an alternating pattern of $c(0.5, -0.5)$ .            |
| seed            | Random seed. Pass NULL to skip <code>set.seed()</code> (useful inside <code>monte_carlo()</code> where the caller manages RNG).      |
| outside_option  | Logical; if TRUE (default) an outside option with <code>alt = 0</code> and zero covariates is added to every choice set.             |
| vary_choice_set | Logical; if TRUE (default) choice set size is sampled uniformly from 2:J; if FALSE every individual faces all J inside alternatives. |

**Value**

A choicer\_sim object.

**Examples**

```
sim <- simulate_mnl_data(N = 1000, J = 5, seed = 123)
print(sim)
```

---

|                   |                                         |
|-------------------|-----------------------------------------|
| simulate_mnp_data | <i>Simulate multinomial probit data</i> |
|-------------------|-----------------------------------------|

---

**Description**

Generates synthetic choice data from the MNP data-generating process estimated by `run_mnprobit()`: latent utility differences against the base alternative (alternative 1),

$$w_i = X_i\beta + \delta + \varepsilon_i, \quad \varepsilon_i \sim N_{J-1}(0, \Sigma),$$

with alternative  $j > 1$  chosen iff  $w_{ij} > \max(0, \max_{k \neq j} w_{ik})$  and the base chosen iff all  $w_{ij} < 0$ . Covariates are Uniform(-1, 1). Choice sets are balanced (every individual faces all J alternatives), as the MNP estimator requires; there is no outside-option flag — model an outside good as a zero-covariate base alternative instead.

**Usage**

```
simulate_mnp_data(
  N = 5000,
  J = 3,
  beta = c(0.8, -0.6),
  delta = NULL,
  Sigma = matrix(c(1, 0.5, 0.5, 1.5), nrow = 2),
  seed = 123
)
```

**Arguments**

|       |                                                                                                                                           |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------|
| N     | Number of choice situations.                                                                                                              |
| J     | Number of alternatives (alternative 1 is the base).                                                                                       |
| beta  | Fixed coefficients for $x_1 \dots x_{K_x}$ (length $K_x = \text{length}(\text{beta})$ ).                                                  |
| delta | ASCs of the differenced utilities, one per non-base alternative (length $J - 1$ ). Defaults to an alternating pattern of $c(0.5, -0.5)$ . |
| Sigma | Covariance matrix of the differenced errors $((J-1) \times (J-1))$ .                                                                      |
| seed  | Random seed (NULL skips <code>set.seed()</code> ).                                                                                        |

### Details

The MNP likelihood only identifies parameters up to scale, so `true_params` is reported on the *identified* scale (normalized by  $\sigma_{11}$ ):  $\text{beta} = \beta/\sqrt{\sigma_{11}}$ ,  $\text{delta} = \delta/\sqrt{\sigma_{11}}$ , and  $\text{Sigma} = \Sigma/\sigma_{11}$  — the scale on which `run_mnprobit()` reports its posterior. With the default Sigma ( $\sigma_{11} = 1$ ) the DGP scale and the identified scale coincide.

### Value

A `choicer_sim` object. `true_params` contains `beta`, `delta`, and `Sigma` on the identified scale (see Details).

### Examples

```
sim <- simulate_mnp_data(N = 1000, J = 3, seed = 123)
print(sim)
```

---

|                                |                                  |
|--------------------------------|----------------------------------|
| <code>simulate_mxl_data</code> | <i>Simulate mixed logit data</i> |
|--------------------------------|----------------------------------|

---

### Description

Generates synthetic choice data with random coefficients drawn from a multivariate normal (optionally log-normal per dimension) and an additional mean shifter `mu`. Random coefficients are parameterized via the lower Cholesky factor of `Sigma`. Covariates are `Uniform(-1, 1)` by default; columns named in `price_cols` are drawn as `-Uniform(0.1, 3)` to mimic strictly-negative price variables. With `T > 1` the data are a panel: each of the `N` decision makers draws one taste vector and keeps it in all of their `T` choice situations (the model of `run_mxlogit(person_col = )`).

### Usage

```
simulate_mxl_data(
  N = 5000,
  J = 4,
  beta = c(0.8, -0.6),
  delta = NULL,
  mu = NULL,
  Sigma = matrix(c(1, 0.5, 0.5, 1.5), nrow = 2),
  rc_dist = NULL,
  rc_correlation = NULL,
  price_cols = NULL,
  seed = 123,
  outside_option = TRUE,
  vary_choice_set = TRUE,
  T = 1L
)
```

## Arguments

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| N               | Number of decision makers; equals the number of choice situations when $T = 1$ .                                                                                                                                                                                                                                                                                                                                                                                                         |
| J               | Number of inside alternatives.                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| beta            | Fixed coefficients for $x_1 \dots x_{K_x}$ (length $K_x = \text{length}(\text{beta})$ ).                                                                                                                                                                                                                                                                                                                                                                                                 |
| delta           | ASCs for inside alternatives (length J). Defaults to an alternating pattern of $c(0.5, -0.5)$ .                                                                                                                                                                                                                                                                                                                                                                                          |
| mu              | Mean shifter for random coefficients (length $K_w = \text{ncol}(\text{Sigma})$ ). Defaults to a zero vector.                                                                                                                                                                                                                                                                                                                                                                             |
| Sigma           | Covariance matrix of random coefficients (square, $K_w \times K_w$ ).                                                                                                                                                                                                                                                                                                                                                                                                                    |
| rc_dist         | Integer vector (length $K_w$ ): 0L for normal, 1L for log-normal. Default NULL is treated as all-normal.                                                                                                                                                                                                                                                                                                                                                                                 |
| rc_correlation  | Logical; if NULL (default) it is auto-detected from the off-diagonal entries of Sigma.                                                                                                                                                                                                                                                                                                                                                                                                   |
| price_cols      | Character vector of $w^*$ column names to draw as $-\text{Uniform}(0.1, 3)$ instead of $\text{Uniform}(-1, 1)$ . Default NULL.                                                                                                                                                                                                                                                                                                                                                           |
| seed            | Random seed (NULL skips <code>set.seed()</code> ).                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| outside_option  | Logical; include outside option with $\text{alt} = 0$ .                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| vary_choice_set | Logical; if TRUE (default) choice set size is sampled uniformly from $2:J$ .                                                                                                                                                                                                                                                                                                                                                                                                             |
| T               | Number of choice situations per decision maker (a single integer $\geq 1$ ; default 1, a cross-section). Situation ids $1 \dots N \times T$ stay globally unique, so <code>id_col = "id"</code> works with or without <code>person_col = "pid"</code> ; with $T > 1$ the data gain a leading <code>pid</code> column identifying decision makers. Choice sets, covariates and shocks are drawn per situation. For a given seed, $T = 1$ reproduces the data of earlier versions exactly. |

## Details

Random coefficients are constructed to match the estimator's parameterization in `src/mxlogit.cpp`. For every dimension the raw draw is  $L \times \eta$  where  $\eta \sim N(0, I)$ . A normal random coefficient (`rc_dist = 0`) is then  $\gamma_k = \mu_k + (L \times \eta)_k$ . A log-normal random coefficient (`rc_dist = 1`) follows the shifted log-normal  $\beta_k = \exp(\mu_k) + \exp((L \times \eta)_k)$  – not the textbook  $\exp(\mu_k + \sigma_k \times \eta)$  – so  $\mu_k$  in `true_params$mu` is on the same scale the estimator recovers and `recovery_table()` can compare like-for-like.

## Value

A `choicer_sim` object. `true_params` includes `beta`, `delta`, `Sigma`, `L_params` (packed Cholesky parameters), `mu`, `rc_dist`, `rc_correlation`, and `gamma_i`: the  $K_w \times N$  matrix of realized random coefficients (rows are the  $w^*$  columns, columns the decision makers), on the scale they enter utility: the realized tastes, which the conditional means from `conditional_tastes()` can be compared with (they track them with shrinkage). `settings` records `T`.

## Examples

```
sim <- simulate_mxl_data(N = 1000, J = 4, seed = 123)
print(sim)
# panel: 200 decision makers x 5 choice situations
panel <- simulate_mxl_data(N = 200, T = 5, J = 4, seed = 123)
head(panel$data)
```

---

|                  |                                   |
|------------------|-----------------------------------|
| simulate_nl_data | <i>Simulate nested logit data</i> |
|------------------|-----------------------------------|

---

## Description

Generates synthetic choice data with nested logit probabilities computed analytically (log-sum-exp over inclusive values), then samples choices from the implied multinomial. The outside option ( $j = 0$ ) sits in a singleton nest with  $\lambda = 1$ .

## Usage

```
simulate_nl_data(
  N = 10000,
  beta = c(1.5, -0.8),
  delta = c(`1` = 0.5, `2` = 0.3, `3` = -0.2, `4` = -0.5, `5` = 0.4),
  nests = list(c(1, 2), c(3, 4, 5)),
  lambdas = c(0.8, 0.2),
  seed = 123
)
```

## Arguments

|         |                                                                           |
|---------|---------------------------------------------------------------------------|
| N       | Number of choice situations.                                              |
| beta    | Fixed coefficients for covariates $X$ , $W$ (length 2 by default).        |
| delta   | Named numeric vector of ASCs for inside alternatives.                     |
| nests   | List of integer vectors defining nest membership for inside alternatives. |
| lambdas | Numeric vector of dissimilarity parameters, one per nest.                 |
| seed    | Random seed (NULL skips <code>set.seed()</code> ).                        |

## Value

A `choicer_sim` object. `true_params` includes `beta`, `delta`, `lambdas`; `settings` includes the `nest_structure`. The returned data retains a `nest` column (integer, with 0L for the outside option) for convenient use with `run_nestlogit()`.

**Note**

Unlike `simulate_mnl_data()` and `simulate_mxl_data()`, this function does not expose `outside_option` or `vary_choice_set` flags. The outside option ( $j = 0$ ) is always present as a singleton nest with  $\lambda = 1$ , and every individual faces the full set of inside alternatives. Add these flags if downstream use cases need them.

**Examples**

```
sim <- simulate_nl_data(N = 2000, seed = 123)
print(sim)
```

---

|                    |                                           |
|--------------------|-------------------------------------------|
| summary.choicer_hb | <i>Summarize a hierarchical Bayes fit</i> |
|--------------------|-------------------------------------------|

---

**Description**

Posterior summaries (mean, SD, equal-tailed credible interval) for the population coefficients  $b$ , the mean-function coefficients  $\theta$ , the alternative-effect variance  $\sigma_d^2$  (and, for the HMNP, the raw shock variance trace), plus the  $\delta_j / \xi_j$  quality ladder, acceptance diagnostics, and a consolidated convergence-diagnostic table (rank-normalized R-hat, ESS bulk/tail, MCSE) built from all retained chains (see `rhat()`, `ess()`, `mcse()`).

**Usage**

```
## S3 method for class 'choicer_hb'
summary(object, prob = 0.95, ...)
```

**Arguments**

|        |                                                                        |
|--------|------------------------------------------------------------------------|
| object | A choicer_hmnl or choicer_hmnp object.                                 |
| prob   | Probability mass of the equal-tailed credible interval (default 0.95). |
| ...    | Additional arguments (ignored).                                        |

**Value**

A summary.choicer\_hb object.

**Examples**

```
sim <- simulate_hmnl_data(N = 50, T = 2, J = 3, seed = 42)
fit <- suppressWarnings(run_hmnllogit(sim$data, "task", "alt", "choice", c("x1", "x2"),
  person_col = "pid",
  mcmc = list(R = 300, burn = 100)))
summary(fit)
```

---

summary.choicer\_mnl      *Summary for multinomial logit model*


---

## Description

Computes and returns a coefficient summary table with standard errors, z-values, p-values, and significance codes. Triggers lazy Hessian computation if standard errors have not been computed yet.

## Usage

```
## S3 method for class 'choicer_mnl'
summary(object, gof = TRUE, ...)
```

## Arguments

|        |                                                                                                                                                                                                               |
|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object | A choicer_mnl object.                                                                                                                                                                                         |
| gof    | Logical; compute goodness-of-fit measures (McFadden R-squared, hit rate) for the summary footer. Involves an in-sample prediction pass (for mixed logit, a full simulation over draws); set to FALSE to skip. |
| ...    | Additional arguments (ignored).                                                                                                                                                                               |

## Value

A summary.choicer\_mnl object (list with coefficients table and metadata, including a gof element with goodness-of-fit measures from [gof](#); its fields are NA when the model was fitted with keep\_data = FALSE).

## Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
summary(fit)
```

---

summary.choicer\_mnp      *Summary for Bayesian multinomial probit model*


---

**Description**

Posterior summaries (mean, SD, equal-tailed credible interval) of the identified coefficient and co-variance draws.

**Usage**

```
## S3 method for class 'choicer_mnp'
summary(object, prob = 0.95, ...)
```

**Arguments**

|        |                                                                        |
|--------|------------------------------------------------------------------------|
| object | A choicer_mnp object.                                                  |
| prob   | Probability mass of the equal-tailed credible interval (default 0.95). |
| ...    | Additional arguments (ignored).                                        |

**Value**

A summary.choicer\_mnp object (list with coefficient and Sigma posterior tables plus metadata).

**Examples**

```
library(data.table)
set.seed(42)
N <- 100; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnpobit(dt, "id", "alt", "choice", c("x1", "x2"),
                  mcmc = list(R = 300, burn = 100))
summary(fit)
```

---

summary.choicer\_mxl      *Summary for mixed logit model*


---

**Description**

Computes coefficient summary with delta-method transformation for variance parameters (Cholesky to covariance scale) and log-normal mean parameters. Triggers lazy Hessian computation if standard errors have not been computed yet.

**Usage**

```
## S3 method for class 'choicer_mxl'
summary(object, gof = TRUE, ...)
```

**Arguments**

|        |                                                                                                                                                                                                               |
|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object | A choicer_mxl object.                                                                                                                                                                                         |
| gof    | Logical; compute goodness-of-fit measures (McFadden R-squared, hit rate) for the summary footer. Involves an in-sample prediction pass (for mixed logit, a full simulation over draws); set to FALSE to skip. |
| ...    | Additional arguments (ignored).                                                                                                                                                                               |

**Value**

A summary.choicer\_mxl object (includes a gof element with goodness-of-fit measures from `gof`; its fields are NA when the model was fitted with `keep_data = FALSE`).

**Examples**

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mxlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = "x1", random_var_cols = "w1", S = 50L
)
summary(fit)
```

---

|                    |                                       |
|--------------------|---------------------------------------|
| summary.choicer_nl | <i>Summary for nested logit model</i> |
|--------------------|---------------------------------------|

---

**Description**

Triggers lazy Hessian computation if standard errors have not been computed yet.

**Usage**

```
## S3 method for class 'choicer_nl'
summary(object, gof = TRUE, ...)
```

Arguments

|        |                                                                                                                                                                                                               |
|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object | A choicer_nl object.                                                                                                                                                                                          |
| gof    | Logical; compute goodness-of-fit measures (McFadden R-squared, hit rate) for the summary footer. Involves an in-sample prediction pass (for mixed logit, a full simulation over draws); set to FALSE to skip. |
| ...    | Additional arguments (ignored).                                                                                                                                                                               |

Value

A summary.choicer\_nl object (includes a gof element with goodness-of-fit measures from [gof](#); its fields are NA when the model was fitted with keep\_data = FALSE).

Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 4
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, nest := ifelse(alt <= 2, "A", "B")]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_nestlogit(
  data = dt, id_col = "id", alt_col = "alt", choice_col = "choice",
  covariate_cols = c("x1", "x2"), nest_col = "nest"
)
summary(fit)
```

---

|             |                                             |
|-------------|---------------------------------------------|
| thread_info | <i>Query choicer OpenMP thread settings</i> |
|-------------|---------------------------------------------|

---

Description

Query choicer OpenMP thread settings

Usage

```
thread_info()
```

Value

A list with OpenMP availability, active/max thread settings, CPU thread capacity reported by OpenMP, thread limits, and relevant environment variables.

---

|           |                                               |
|-----------|-----------------------------------------------|
| traceplot | <i>Traceplot for a hierarchical Bayes fit</i> |
|-----------|-----------------------------------------------|

---

**Description**

Generic dispatching on the fit's class. See `traceplot.choicer_hb()` for the `choicer_hmnl` / `choicer_hmnp` method.

**Usage**

```
traceplot(object, ...)
```

**Arguments**

|                     |                                         |
|---------------------|-----------------------------------------|
| <code>object</code> | A fitted model object.                  |
| <code>...</code>    | Additional arguments passed to methods. |

**Value**

The object, invisibly.

---

|                                   |                                                     |
|-----------------------------------|-----------------------------------------------------|
| <code>traceplot.choicer_hb</code> | <i>Traceplot method for hierarchical Bayes fits</i> |
|-----------------------------------|-----------------------------------------------------|

---

**Description**

Overlaid per-chain traceplots (base graphics, no new dependency) for a `choicer_hmnl` / `choicer_hmnp` fit's population coefficients (`b`), delta mean-function coefficients (`theta`), alternative-effect variance (`sigma_d2`), and, opt-in, a representative subset of the (potentially ~200-column) alternative effects (`delta`).

**Usage**

```
## S3 method for class 'choicer_hb'
traceplot(object, block = c("b", "theta", "sigma_d2"), which = NULL, ...)
```

**Arguments**

|                     |                                                                                                                                                                                                                                                                                                                                    |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>object</code> | A <code>choicer_hmnl</code> or <code>choicer_hmnp</code> fit.                                                                                                                                                                                                                                                                      |
| <code>block</code>  | Character vector, any non-empty subset of <code>c("b", "theta", "sigma_d2", "delta")</code> (HMNP fits additionally accept <code>"sigma2"</code> , the raw non-identified shock-variance chain). Default plots <code>b</code> , <code>theta</code> , <code>sigma_d2</code> (not <code>delta</code> , which can have ~200 columns). |

|       |                                                                                                                                                                                                                                                                                       |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| which | Only consulted when "delta" is in block. NULL (default) auto-selects a representative subset (the 3 highest rank-R-hat plus 3 lowest ESS_bulk alternatives, deduplicated, capped at 6). If supplied, a character vector of alternative labels or an integer vector of column indices. |
| ...   | Additional arguments (ignored).                                                                                                                                                                                                                                                       |

### Value

object, invisibly.

### Examples

```
sim <- simulate_hmnl_data(N = 60, T = 2, J = 4, seed = 1)
fit <- suppressWarnings(run_hmnl_logit(sim$data, "task", "alt", "choice", c("x1", "x2"),
  person_col = "pid",
  mcmc = list(R = 300, burn = 100), chains = 2))
traceplot(fit)
```

---

|                  |                                                                     |
|------------------|---------------------------------------------------------------------|
| vcov.choicer_fit | <i>Extract variance-covariance matrix from a choicer_fit object</i> |
|------------------|---------------------------------------------------------------------|

---

### Description

With no arguments, returns the variance-covariance matrix implied by the fit's own `se_method` (triggering lazy computation if needed). Passing `type` recomputes a different variance estimator post hoc from the stored data — no refit needed (requires `keep_data = TRUE`):

"hessian" Inverse of the analytical negated Hessian (the numerical one for a nested logit fitted with `se_method = "numeric"`).

"bhhh" Inverse of the BHHH/OPG information  $\sum_i w_i s_i s_i'$ .

"robust" Huber-White sandwich  $A^{-1}(\sum_i w_i^2 s_i s_i')A^{-1}$  — also the valid WESML variance under choice-based weighting.

"cluster" Cluster-robust sandwich  $A^{-1}(\sum_g g_g g_g')A^{-1}$  with  $g_g = \sum_{i \in g} w_i s_i$  the within-cluster sum of weighted scores. Requires `cluster` (or a fit made with `cluster_col`). No small-sample correction is applied.

Here  $i$  indexes *likelihood units*: choice situations, or decision makers for a mixed logit fitted with `person_col` (panel likelihood). For repeated choices by the same decision maker in a cross-sectional fit, cluster on the decision maker.

### Usage

```
## S3 method for class 'choicer_fit'
vcov(object, type = NULL, cluster = NULL, ...)
```

## Arguments

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object  | A choicer_fit object.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| type    | NULL (default; return the as-fitted vcov) or one of "hessian", "bhhh", "robust", "cluster".                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| cluster | Cluster labels for type = "cluster", one per choice situation (collapsed to one per decision maker for a panel mixed logit fit). Alignment to the prepared (id-sorted) choice situations is handled as follows: <ul style="list-style-type: none"> <li>• <b>Named</b> (recommended): names are matched against the choice-situation ids, so the vector is safe in any order. Build it by naming your per-situation labels with the id values.</li> <li>• <b>Unnamed</b>: taken to be in the prepared order (ascending id), with a warning; for a panel fit whose prepared order (decision maker, then id) differs from id order, an unnamed vector is rejected — name it by choice-situation id or use cluster_col.</li> </ul> Defaults to the labels stored at fit time via cluster_col (already aligned). Supplying cluster without type implies type = "cluster". The safest route is to pass cluster_col= at fit time, which sidesteps post-hoc alignment entirely. |
| ...     | Additional arguments (ignored).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## Details

Note (mixed logit): without person\_col, clustering repairs the *inference*, not the *likelihood*. The cross-sectional likelihood treats each choice situation as an independent draw from the mixing distribution; on panel data it is a composite likelihood that still targets the population taste distribution  $f(\beta \mid \theta)$ , just inefficiently. type = "cluster" makes the point estimates' standard errors robust to within-person dependence but does not recover the efficiency of the panel likelihood (person\_col). With person\_col the scores are per decision maker, so type = "robust" is already clustered by decision maker; cluster labels must then nest decision makers (be constant within each), and type = "cluster" adds something only for coarser groups such as households or markets.

## Value

Named variance-covariance matrix, or NULL if unavailable: when the information is singular, or when the variance failed at fit time (the fit warned, and its variance\_error holds the error; see [run\\_mnlogit](#)). For such a fit, type computes the variance from the stored data and returns it; the fit itself keeps its NA standard errors. Removing the record, fit\$variance\_error <- NULL, lets summary() and the other methods compute the variance each time they need it.

## Examples

```
library(data.table)
set.seed(42)
N <- 50; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, person := rep(1:10, each = 5)[id]]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
```

```

fit <- run_mnlogit(dt, "id", "alt", "choice", c("x1", "x2"))
vcov(fit)                                # as fitted (hessian)
vcov(fit, type = "robust")               # Huber-White, post hoc
# named by situation id -> safe regardless of order
cl <- dt[, person[1L], by = id]
vcov(fit, type = "cluster", cluster = setNames(cl$V1, cl$id))

```

---

|                 |                                                            |
|-----------------|------------------------------------------------------------|
| vcov.choicer_hb | <i>Posterior covariance of the population coefficients</i> |
|-----------------|------------------------------------------------------------|

---

### Description

Posterior covariance of the population coefficients

### Usage

```

## S3 method for class 'choicer_hb'
vcov(object, ...)

```

### Arguments

|        |                                        |
|--------|----------------------------------------|
| object | A choicer_hmnl or choicer_hmnp object. |
| ...    | Additional arguments (ignored).        |

### Value

K x K posterior covariance matrix of the b draws.

---

|                  |                                                                     |
|------------------|---------------------------------------------------------------------|
| vcov.choicer_mnp | <i>Extract variance-covariance matrix from a choicer_mnp object</i> |
|------------------|---------------------------------------------------------------------|

---

### Description

Returns the posterior covariance matrix of the identified coefficient draws (computed eagerly at fit time; no Hessian is involved).

### Usage

```

## S3 method for class 'choicer_mnp'
vcov(object, ...)

```

### Arguments

|        |                                 |
|--------|---------------------------------|
| object | A choicer_mnp object.           |
| ...    | Additional arguments (ignored). |

**Value**

Named posterior covariance matrix.

**Examples**

```
library(data.table)
set.seed(42)
N <- 100; J <- 3
dt <- data.table(id = rep(1:N, each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), x2 = rnorm(.N))]
dt[, choice := 0L]
dt[, choice := sample(c(1L, rep(0L, J - 1))), by = id]
fit <- run_mnpobit(dt, "id", "alt", "choice", c("x1", "x2"),
                  mcmc = list(R = 300, burn = 100))
vcov(fit)
```

---

wesml\_vcov

*Robust (sandwich) variance for a weighted / choice-based logit fit*

---

**Description**

Recomputes the robust Huber-White sandwich variance  $V = A^{-1}BA^{-1}$  for a fitted multinomial (MNL), mixed (MXL) or nested (NL) logit, where the bread  $A = \sum_i w_i(-H_i)$  is the weighted negated Hessian and the meat  $B = \sum_i w_i^2 s_i s_i'$  is the weight-squared outer product of the per-individual scores. This is the appropriate variance under choice-based (endogenous stratified) / WESML weighting, where the inverse-Hessian and the ordinary BHHH variance are invalid. It can be called on any fitted model (e.g. one estimated with `se_method = "hessian"`) to obtain robust standard errors post hoc, without refitting.

**Usage**

```
wesml_vcov(object, ...)

## S3 method for class 'choicer_mxl'
wesml_vcov(object, type = c("vcov", "se"), ...)

## S3 method for class 'choicer_mnl'
wesml_vcov(object, type = c("vcov", "se"), ...)

## S3 method for class 'choicer_nl'
wesml_vcov(object, type = c("vcov", "se"), ...)
```

**Arguments**

|                     |                                                                                                                                           |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <code>object</code> | A fitted <code>choicer_mnl</code> , <code>choicer_mxl</code> or <code>choicer_nl</code> object (requires <code>keep_data = TRUE</code> ). |
| <code>...</code>    | Unused.                                                                                                                                   |

type Either "vcov" (default) to return the variance-covariance matrix or "se" to return the standard-error vector.

### Details

If the stored weights are uniform (all equal), a warning is emitted: the returned variance is then the ordinary robust (Huber-White) variance, not a WESML-weighted variance. Refit with WESML weights for a choice-based-sampling correction.

### Value

A variance-covariance matrix (type = "vcov") or a named numeric vector of standard errors (type = "se"), in the raw parameter space.

### See Also

[wesml\\_weights](#), [sample\\_by\\_choice](#), [run\\_mxlogit](#)

### Examples

```
library(data.table)
set.seed(1)
N <- 200L; J <- 3L
dt <- data.table(id = rep(seq_len(N), each = J), alt = rep(1:J, N))
dt[, `:=`(x1 = rnorm(.N), w1 = rnorm(.N))]
dt[, choice := as.integer(seq_len(.N) == sample.int(.N, 1L)), by = id]
fit <- run_mxlogit(dt, "id", "alt", "choice", "x1", "w1", S = 50L)
wesml_vcov(fit, "se")
```

---

wesml\_weights

*WESML weights for choice-based (endogenous stratified) samples*

---

### Description

Computes Manski-Lerman (1977) Weighted Exogenous Sample Maximum Likelihood (WESML) weights for a choice-based sample. The weight for a choice situation whose chosen alternative is  $j$  is  $w = Q(j)/H(j)$ , where  $Q(j)$  is the population share of alternative  $j$  and  $H(j)$  its sample share among choosers. Using these weights in [run\\_mxlogit](#) restores consistency under choice-based sampling; pair them with `se_method = "sandwich"` for valid (robust) standard errors (the plain inverse-Hessian is invalid under weighting).

### Usage

```
wesml_weights(
  data,
  id_col,
  alt_col,
  choice_col,
```

```

    Q,
    H = NULL,
    normalize = TRUE,
    attach = FALSE,
    weight_name = ".wesml_weight",
    outside_opt_label = NULL,
    include_outside_option = FALSE
  )

```

## Arguments

|                                                        |                                                                                                                                                                                                                                                         |
|--------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code>                                      | A long-format choice data set (data.frame or data.table), one row per alternative per choice situation.                                                                                                                                                 |
| <code>id_col, alt_col, choice_col</code>               | Column names identifying the choice situation, the alternative, and the 0/1 chosen indicator.                                                                                                                                                           |
| <code>Q</code>                                         | Named numeric vector of population shares, one entry per chosen stratum (names matched to <code>as.character(alt)</code> ), each strictly positive. Renormalized to sum 1 if needed.                                                                    |
| <code>H</code>                                         | Optional named numeric vector of sample shares. If NULL (default) it is computed from data as the fraction of choice situations choosing each alternative.                                                                                              |
| <code>normalize</code>                                 | If TRUE (default) the returned weights are scaled to mean 1. This does not affect the point estimates or the sandwich variance.                                                                                                                         |
| <code>attach</code>                                    | If TRUE, return data with a row-level weight column appended (the per-situation weight repeated across all rows of a situation), ready to pass to <code>run_mxlogit(weights_col = ...)</code> . If FALSE (default) return an id-keyed table of weights. |
| <code>weight_name</code>                               | Name of the weight column (default <code>".wesml_weight"</code> ).                                                                                                                                                                                      |
| <code>outside_opt_label, include_outside_option</code> | Set <code>include_outside_option = TRUE</code> and supply <code>outside_opt_label</code> when the outside good is implicit (choice situations with no 1 in <code>choice_col</code> are treated as having chosen the outside good).                      |

## Details

Strata are defined by the chosen alternative and keyed by `as.character(alt)` so numeric and character alternative codes match supplied share names unambiguously.

## Value

Either an id-keyed data.table with columns `id_col` and `weight_name` (default), or, when `attach = TRUE`, a copy of data with the weight column appended. The result carries "Q", "H", and "choice\_sampling" attributes recording provenance.

## References

Manski, C. F. and Lerman, S. R. (1977). The Estimation of Choice Probabilities from Choice Based Samples. *Econometrica* 45(8), 1977-1988. Train, K. E. (2009). *Discrete Choice Methods with Simulation*, Section 3.7. Cambridge University Press.

**See Also**

[sample\\_by\\_choice](#), [run\\_mxlogit](#), [wesml\\_vcov](#)

**Examples**

```
library(data.table)
set.seed(1)
N <- 300L; J <- 3L
pop <- data.table(id = rep(seq_len(N), each = J), alt = rep(1:J, N))
pop[, x1 := rnorm(.N)]
pop[, w1 := rnorm(.N)]
pop[, choice := as.integer(seq_len(.N) == sample.int(.N, 1L)), by = id]

# Population shares of the chosen alternative
Q <- prop.table(table(pop[choice == 1, alt]))
wt <- wesml_weights(pop, "id", "alt", "choice", Q = Q)
head(wt)
```

---

wtp.choicer\_hb

---

*Compute willingness to pay*


---

**Description**

Computes willingness-to-pay (WTP) ratios with delta-method standard errors from a fitted choice model. For an attribute coefficient  $\theta_k$  and a price coefficient  $\theta_p$ , the WTP is

$$WTP_k = -\theta_k/\theta_p,$$

the marginal rate of substitution between the attribute and price. Standard errors use the delta method with analytic gradients  $\partial g/\partial\theta_k = -1/\theta_p$  and  $\partial g/\partial\theta_p = \theta_k/\theta_p^2$ , applied to the corresponding 2x2 block of `vcov(object)`.

**Usage**

```
## S3 method for class 'choicer_hb'
wtp(object, price_var, attr_vars = NULL, level = 0.95, ...)

wtp(object, price_var, attr_vars = NULL, level = 0.95, ...)

## S3 method for class 'choicer_fit'
wtp(object, price_var, attr_vars = NULL, level = 0.95, ...)

## S3 method for class 'choicer_mxl'
wtp(object, price_var, attr_vars = NULL, level = 0.95, ...)
```

**Arguments**

|           |                                                                                                                                                                                                                                                                          |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| object    | A fitted model object (choicer_mnl, choicer_mxl, or choicer_nl).                                                                                                                                                                                                         |
| price_var | Name of the price variable. Must be a fixed-coefficient variable (a column of the design matrix X).                                                                                                                                                                      |
| attr_vars | Character vector of attributes to report. Defaults to all fixed-coefficient variables other than price_var (plus, for mixed logit with rc_mean = TRUE, all random coefficients). ASC names (e.g. "ASC_2") may also be supplied; the WTP of an ASC is $-ASC_j/\theta_p$ . |
| level     | Confidence level for the normal-approximation interval $Estimate \pm z_{1-(1-level)/2} \times SE$ . Default 0.95.                                                                                                                                                        |
| ...       | Additional arguments passed to methods.                                                                                                                                                                                                                                  |

**Details**

For mixed logit models, random coefficients are included via their estimated location parameters. The package's log-normal random coefficient is the *shifted* log-normal  $\beta_k = \exp(\mu_k) + \exp((L\eta)_k)$  (see `run_mxlogit()`), so:

- Normal random coefficient  $k$  (`rc_mean = TRUE`): mean WTP  $-\mu_k/\theta_p$ , labeled `Mu_x`.
- Log-normal random coefficient  $k$  (`rc_mean = TRUE`): **median** WTP  $-(\exp(\mu_k) + 1)/\theta_p$ , since the median of  $\exp((L\eta)_k)$  is 1. (The mean,  $\exp(\mu_k) + \exp(\sigma_k^2/2)$ , is highly sensitive to the estimated variance; the median is the more robust summary.) These rows are labeled by the attribute name and flagged as medians when printed.
- Log-normal random coefficient with `rc_mean = FALSE`:  $\beta_k = \exp((L\eta)_k)$  has median 1, so the median WTP is  $-1/\theta_p$  with uncertainty driven solely by  $\theta_p$ .

Normal random coefficients with `rc_mean = FALSE` have mean 0 by construction and are excluded from the table.

The price variable must have a *fixed* coefficient. A random price coefficient is rejected: the ratio of two random coefficients generally has no finite moments (the denominator has positive density at 0), so mean or median WTP computed from location parameters would be meaningless. In `choicer`, use a fixed price coefficient. WTP-space estimation is not currently implemented; it is an alternative specification available in other software rather than an option supplied by this function.

**Value**

A `data.frame` of class `choicer_wtp` with one row per attribute and columns `Estimate`, `Std_Error`, `z_value`, `CI_lower`, `CI_upper`. Attributes `price_var` and `level` record the inputs; `median_rows` lists rows that are median (rather than mean) WTP. Standard errors are NA when the variance-covariance matrix is unavailable.

**Methods (by class)**

- `wtp(choicer_hb)`: Posterior willingness-to-pay for hierarchical Bayes fits: the per-draw ratio of population-mean utility coefficients,  $-\bar{\gamma}_{attr}/\bar{\gamma}_{price}$  (for log-normal coordinates  $\bar{\gamma} = \exp(b + W_{kk}/2)$ ). Ratio posteriors are heavy-tailed, so the point estimate is the posterior **median** with equal-tailed quantile intervals — never a posterior mean or a delta-method SE. A

warning is raised when the price coefficient's sign is not resolved by the posterior. If the price variable was flagged as endogenous-without-a-control-function at prep time, WTP inherits that caveat (see `cf_residual_col` in [prepare\\_hmnl\\_data\(\)](#)).

### Examples

```
library(data.table)
sim <- simulate_mnl_data(N = 1000, J = 4, beta = c(0.8, -0.6), seed = 123,
                        outside_option = FALSE, vary_choice_set = FALSE)
fit <- run_mnlogit(sim$data, "id", "alt", "choice", c("x1", "x2"))
# treat x2 as the price variable
wtp(fit, price_var = "x2")
wtp(fit, price_var = "x2", attr_vars = c("x1", "ASC_2"), level = 0.90)
```

# Index

## \* datasets

- mode\_choice, 33
- blp, 4, 36
- blp.choicer\_mnl, 5
- blp.choicer\_mxl, 6, 36
- blp.choicer\_nl, 7
- blp\_contraction, 8
- coef.choicer\_fit, 10
- coef.choicer\_hb, 10
- coef.choicer\_mnp, 11
- conditional\_tastes, 12, 64, 83, 86, 88, 90
- conditional\_tastes(), 46, 102
- consumer\_surplus, 30
- consumer\_surplus
  - (consumer\_surplus.choicer\_hmnl), 14
- consumer\_surplus.choicer\_hmnl, 14
- diversion\_ratios
  - (diversion\_ratios.choicer\_hb), 17
- diversion\_ratios.choicer\_hb, 17
- diversion\_ratios.choicer\_mnl, 18
- diversion\_ratios.choicer\_mxl, 19
- diversion\_ratios.choicer\_nl, 20
- elasticities (elasticities.choicer\_hb), 21
- elasticities.choicer\_hb, 21
- elasticities.choicer\_hb(), 18
- elasticities.choicer\_mnl, 22
- elasticities.choicer\_mxl, 23
- elasticities.choicer\_nl, 24
- ess, 25, 32
- ess(), 72, 73, 75, 76, 104
- fft, 25
- get\_halton\_normals, 7, 25, 36, 83, 86, 88
- gof, 26, 105, 107, 108
- logLik.choicer\_fit, 27
- logsum, 14, 16, 17
- logsum(logsum.choicer\_hmnl), 28
- logsum.choicer\_hmnl, 28
- logsum.choicer\_hmnp(), 17
- mc\_asymptotics, 30
- mcse, 32
- mcse(), 72, 73, 75, 76, 104
- mode\_choice, 33
- monte\_carlo, 34
- monte\_carlo(), 30
- mxl\_blp\_contraction, 35
- mxl\_logsum, 28
- new\_choicer\_sim, 37
- nl\_blp\_contraction, 38
- nobs.choicer\_fit, 39
- nobs.choicer\_hb, 40
- nobs.choicer\_mnp, 40
- ppc\_shares, 41
- predict.choicer\_hb, 42
- predict.choicer\_hb(), 41
- predict.choicer\_mnl, 43
- predict.choicer\_mxl, 45
- predict.choicer\_nl, 46
- prepare\_hmnl\_data, 48
- prepare\_hmnl\_data(), 50–52, 72, 73, 97, 118
- prepare\_hmnp\_data, 50
- prepare\_hmnp\_data(), 50, 75, 76, 98
- prepare\_mnl\_data, 52, 58, 60, 77, 79
- prepare\_mnl\_data(), 49
- prepare\_mnp\_data, 54, 80, 81
- prepare\_mnp\_data(), 51
- prepare\_mxl\_data, 56, 83, 86–88
- prepare\_nl\_data, 58, 93

print.choicer\_cs, 60  
 print.choicer\_fit, 61  
 print.choicer\_gof, 62  
 print.choicer\_hb, 62  
 print.choicer\_mnp, 63  
 print.choicer\_tastes, 64  
 print.choicer\_wtp, 64  
 print.summary.choicer\_hb, 65  
 print.summary.choicer\_mnl, 65  
 print.summary.choicer\_mnp, 66  
 print.summary.choicer\_mxl, 67  
 print.summary.choicer\_nl, 67  
  
 recovery\_table, 68  
 recovery\_table(), 37, 73  
 rhat, 70  
 rhat(), 73, 104  
 run\_hmnlogit, 12, 71, 90  
 run\_hmnlogit(), 75, 76  
 run\_hmnprobit, 74  
 run\_mnlogit, 76, 84, 91, 111  
 run\_mnprobit, 79  
 run\_mnprobit(), 100, 101  
 run\_mxlogit, 14, 26, 82, 95, 114, 116  
 run\_nestlogit, 90  
 run\_nestlogit(), 103  
  
 sample\_by\_choice, 87, 93, 114, 116  
 set\_num\_threads, 88, 95  
 simulate\_hmnl\_data, 96  
 simulate\_hmnl\_data(), 73, 97  
 simulate\_hmnp\_data, 97  
 simulate\_hmnp\_data(), 70, 76  
 simulate\_mnl\_data, 99  
 simulate\_mnl\_data(), 37, 104  
 simulate\_mnp\_data, 100  
 simulate\_mnp\_data(), 69  
 simulate\_mxl\_data, 14, 101  
 simulate\_mxl\_data(), 37, 104  
 simulate\_nl\_data, 103  
 simulate\_nl\_data(), 37  
 summary.choicer\_hb, 104  
 summary.choicer\_mnl, 105  
 summary.choicer\_mnp, 106  
 summary.choicer\_mxl, 106  
 summary.choicer\_nl, 107  
  
 thread\_info, 108  
 thread\_info(), 95  
  
 traceplot, 109  
 traceplot(), 72, 73, 75, 76  
 traceplot.choicer\_hb, 109  
 traceplot.choicer\_hb(), 109  
  
 vcov.choicer\_fit, 110  
 vcov.choicer\_hb, 112  
 vcov.choicer\_mnp, 112  
  
 wesml\_vcov, 83, 86, 113, 116  
 wesml\_weights, 33, 87, 94, 95, 114, 114  
 wtp, 16, 17  
 wtp(wtp.choicer\_hb), 116  
 wtp.choicer\_hb, 116  
 wtp.choicer\_hb(), 17