Package: nrMCmix (via r-universe)

October 22, 2026

Type Package

Title Non-Reversible Markov Chains for Bayesian Mixture Models

Version 0.3

Date 2026-10-13

Description Provides Bayesian inference for finite and infinite mixture models using Markov chain Monte Carlo (MCMC) methods. Supports Gaussian, multivariate Gaussian, Poisson, Binomial, and multivariate Bernoulli mixture kernels with different covariance structures and flexible prior specifications. Provides tools for data generation, posterior density evaluation, clustering and similarity analysis, and advanced MCMC sampling strategies, including reversible and non-reversible samplers: Ascolani et al. (2026) <doi:10.48550/arXiv.2510.03226>, informed proposals: Zanella (2020) <doi:10.1080/01621459.2019.1585255>, collapsed Gibbs sampling: Diebolt and Robert (1994) <doi:10.1111/j.2517-6161.1994.tb01985.x>, and split-and-merge samplers: Dahl and Newcomb (2021) <doi:10.1080/00949655.2021.1998502>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1.0)

Imports base, coda, graphics, grDevices, fields, ggplot2, salso, Rcpp (>= 1.0.12), Rdpack, stats

RdMacros Rdpack

LinkingTo Rcpp, RcppArmadillo

Suggests knitr, rmarkdown, bmixture, datasets

RoxygenNote 7.3.1

VignetteBuilder knitr

NeedsCompilation yes

Author Filippo Ascolani [aut] (ORCID: <https://orcid.org/0009-0009-3899-8877>), Paolo Manildo [aut, cre] (ORCID: <https://orcid.org/0009-0006-7056-7012>), Giacomo Zanella [aut] (ORCID: <https://orcid.org/0000-0001-6727-9884>)

Maintainer Paolo Manildo <paolo.manildo@studenti.unipd.it>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-22 15:24:36 UTC

RemoteUrl https://github.com/cran/nrMCmix

RemoteRef HEAD

RemoteSha 4c7dcc5d817d6c702b24c3e6e49f0dbf9e9fb6dc

Contents

nrMCmix-package . . . 2
compute_similarity_matrix . . . 5
extract_params . . . 6
generate_data . . . 7
nrMCmix . . . 8
plot.nrMCmix . . . 9
posterior_density1 . . . 13
posterior_density2 . . . 14
print.nrMCmix . . . 15
print.summary.nrMCmix . . . 15
priorK . . . 16
specify_model . . . 17
specify_sampler . . . 19
summary.nrMCmix . . . 20

Index **22**

`nrMCmix-package` *Non-Reversible Markov Chains for Bayesian Mixture Models*

Description

Provides Bayesian inference for finite and infinite mixture models using Markov chain Monte Carlo (MCMC) methods. Supports Gaussian, multivariate Gaussian, Poisson, Binomial, and multivariate Bernoulli mixture kernels with different covariance structures and flexible prior specifications. Provides tools for data generation, posterior density evaluation, clustering and similarity analysis, and advanced MCMC sampling strategies, including reversible and non-reversible samplers: Ascolani et al. (2026) <doi:10.48550/arXiv.2510.03226>, informed proposals: Zanella (2020) <doi:10.1080/01621459.2019.1585255>, collapsed Gibbs sampling: Diebolt and Robert (1994) <doi:10.1111/j.2517-6161.1994.tb01985.x>, and split-and-merge samplers: Dahl and Newcomb (2021) <doi:10.1080/00949655.2021.1998502>.

Details

The DESCRIPTION file:

Package:	nrMCmix
Type:	Package
Title:	Non-Reversible Markov Chains for Bayesian Mixture Models
Version:	0.3
Date:	2026-10-13
Authors@R:	c(person("Filippo", "Ascolani", email = "filippo.ascolani@duke.edu", role = "aut", comment = c(ORCID =
Description:	Provides Bayesian inference for finite and infinite mixture models using Markov chain Monte Carlo (MCM
License:	MIT + file LICENSE
Encoding:	UTF-8
Depends:	R (>= 4.1.0)
Imports:	base, coda, graphics, grDevices, fields, ggplot2, salso, Rcpp (>= 1.0.12), Rdpack, stats
RdMacros:	Rdpack
LinkingTo:	Rcpp, RcppArmadillo
Suggests:	knitr, rmarkdown, bmixture, datasets
RoxygenNote:	7.3.1
VignetteBuilder:	knitr
Author:	Filippo Ascolani [aut] (<https://orcid.org/0009-0009-3899-8877>), Paolo Manildo [aut, cre] (<https://orcid
Maintainer:	Paolo Manildo <paolo.manildo@studenti.unipd.it>

Index of help topics:

```
compute_similarity_matrix
                        Compute the posterior similarity matrix
extract_params          Extract a Parameter from a 'nrMCmix' object.
generate_data           Simulate data from a Bayesian mixture-model
                        generative process.
nrMCmix                 MCMC inference for Bayesian mixture models
nrMCmix-package         Non-Reversible Markov Chains for Bayesian
                        Mixture Models
plot.nrMCmix            Plot diagnostics for an 'nrMCmix' model
posterior_density1      Evaluate the posterior predictive density
posterior_density2      Evaluate the bivariate posterior predictive
                        density
print.nrMCmix           Print an 'nrMCmix' model fit
print.summary.nrMCmix   Print an 'nrMCmix' model summary
priorK                  Prior distribution of the number of clusters
specify_model           Specify the model and prior distributions
specify_sampler         Specify the sampling algorithm
summary.nrMCmix         Summarize an nrMCmix model fit
```

The package provides functions for inference on Bayesian mixture models with conjugate priors using a variety of marginal MCMC schemes. The number of mixture components can either be fixed a priori or inferred using a Pitman–Yor mixture model. The package primarily implements

the non-reversible methods developed by Ascolani et al. (2026), while also providing the split-and-merge methods of Dahl and Newcomb (2022) and novel random-neighborhood informed proposals for mixture models, inspired by Zanella (2020). These three families of algorithms can be combined to construct flexible sampling schemes for Bayesian mixture models.

Author(s)

Filippo Ascolani [aut] (<https://orcid.org/0009-0009-3899-8877>), Paolo Manildo [aut, cre] (<https://orcid.org/0009-0006-7056-7012>), Giacomo Zanella [aut] (<https://orcid.org/0000-0001-6727-9884>)

Maintainer: Paolo Manildo <paolo.manildo@studenti.unipd.it>

References

Ascolani, F. Manildo, P. and Zanella, G. (2026). *Fast non-reversible samplers for Bayesian mixture models*. doi:10.48550/arXiv.2510.03226.

Dahl, D. B. and Newcomb, S. (2021). *Sequentially allocated merge-split samplers for conjugate Bayesian nonparametric models*. doi:10.1080/00949655.2021.1998502.

Zanella, G. (2020). *Informed proposals for discrete state-space models*. doi:10.1080/01621459.2019.1585255.

Examples

```
#load the iris data
data(iris)

#fit the model
set.seed(132)
fit <- nrMCmix(as.matrix(iris[1:4]),
               K = 0,
               model = specify_model("MGaussian", dense = TRUE,
                                     alpha0 = 4.01,
                                     hyperpar_baseline = TRUE),
               sampler = specify_sampler())

#check the model fit
{
  op <- par(no.readonly = TRUE)
  par(mfrow = c(4,4), mar = c(1.1,1.1,1.1,1.1))
  for(i in 1:4){
    for(j in 1:4){
      if(i == j){
        #univariate plot
        plot(fit,idx1 = i, verbose = FALSE, gg = FALSE)
      }else{
        #bivariate plot
        plot(fit,idx1 = i, idx2 = j, verbose = FALSE, gg = FALSE)
      }
    }
  }
  par(op)
}
```

```
#check the summary
summary(fit)

#plot the number of components distribution
plot(fit,5,par_name = "K")
```

---

compute_similarity_matrix

Compute the posterior similarity matrix

Description

Computes the posterior similarity matrix from sampled allocation configurations. Each entry gives the proportion of MCMC iterations in which two units are assigned to the same mixture component.

Usage

```
compute_similarity_matrix(c)
```

Arguments

c	Integer matrix of allocation configurations, with one row per MCMC iteration and one column per unit.

Value

A numeric n by n symmetric matrix, where the (i, j) entry is the proportion of sampled configurations in which units i and j belong to the same component.

Examples

```
#specify the model
model <- specify_model("Gaussian", sigma2 = 1, lambda0 = 0.1)

#generate some data
set.seed(123)
y <- generate_data(n = 100, K = 7, model = model)$y

#run a non-reversible split and merge MCMC for a DP mixture
fit <- nrMCmix(y, K = 0, model = model,
               sampler = specify_sampler(SAM = TRUE))

#compute the similarity matrix
S <- compute_similarity_matrix(fit$c)
```

extract_params *Extract a Parameter from a* nrMCmix *object.*

Description

Extracts a parameter or a transformation of the parameters, from the output of the model-fitting procedure of function nrMCmix.

Usage

```
extract_params(
  X,
  par_name = "log_p",
  lab_name = NULL,
  which_coordinate = 1,
  f = function(x) x
)
```

Arguments

X	An object of class nrMCmix, output of the function nrMCmix.
par_name	A character string specifying the name of the parameter to extract.
lab_name	An optional character string specifying the name of the parameter in the returned data frame. If NULL, a name is generated automatically.
which_coordinate	A numeric index specifying which coordinate of the parameter to extract. For vector-type parameters, it is a scalar, while for matrix-type parameters, it can be also a vector of length 2.
f	A function used to transform the parameter before extracting the requested coordinate.

Value

A data frame containing the extracted parameter. The returned column is named according to lab_name. If lab_name is NULL, the name is generated from par_name and which_coordinate.

Examples

```
#specify the model
model <- specify_model("Gaussian", sigma2 = 1, lambda0 = 0.1)

#generate some data
set.seed(123)
y <- generate_data(n = 100, K = 7, model = model)$y

#run a non-reversible split and merge MCMC for a DP mixture
fit <- nrMCmix(y, K = 0, model = model,
```

```
 sampler = specify_sampler(SAM = TRUE))

#extract the posterior draws of the largest cluster proportion
largest_cluster <- extract_params(fit,"c","n1",f = function(x) max(table(x)) / length(x))
```

---

generate_data *Simulate data from a Bayesian mixture-model generative process.*

Description

Simulate data from a Bayesian mixture-model generative process.

Usage

```
generate_data(n = 100L, K = 0L, model = NULL, D = 2L, xi = 100)
```

Arguments

n	Number of observations to simulate.
K	Number of mixture components. Set to 0 for an infinite mixture model.
model	Optional model specification created by specify_model.
D	Number of dimensions for multivariate models.
xi	Parameter controlling the number of trials in the Binomial model.

Value

A list containing y, the simulated observations, and groups, the corresponding mixture-component assignments.

Examples

```
#specify the model
model <- specify_model("Gaussian", sigma2 = 1, lambda0 = 0.1)

#generate some data
set.seed(123)
y <- generate_data(n = 100, K = 7, model = model)$y
```

nrMCmix *MCMC inference for Bayesian mixture models*

Description

Performs Markov chain Monte Carlo inference for Bayesian finite or nonparametric mixture models.

Usage

```
nrMCmix(
  y = NULL,
  K = 0L,
  model = NULL,
  sampler = NULL,
  init = NULL,
  N = 1000L,
  warm_up = 100L,
  thin = 1L,
  thin_scan = 0L,
  refresh_console = 0.1,
  save_parameters = TRUE,
  save_configurations = TRUE
)
```

Arguments

`y`	Data vector or matrix.
`K`	Number of mixture components. If `K > 0`, a finite mixture with `K` components is fitted; if `K = 0`, a nonparametric mixture model is used.
`model`	Optional model specification produced by `specify_model`. If `NULL`, the default model specification is used.
`sampler`	Optional sampler specification produced by `specify_sampler`. If `NULL`, the default sampler specification is used.
`init`	Optional initial allocation vector. If `NULL`, the configuration is initialized randomly for finite mixtures and with a single component for nonparametric mixtures.
`N`	Number of MCMC iterations saved after warm-up.
`warm_up`	Number of warm-up iterations.
`thin`	Thinning interval for the saved MCMC iterations.
`thin_scan`	Number of scans between recorded iterations. If `0`, it is set to the sample size.
`refresh_console`	Fraction of the MCMC progress after which the console is refreshed.

`save_parameters`
: Logical; if `TRUE`, save the sampled model parameters.

`save_configurations`
: Logical; if `TRUE`, save all sampled allocation configurations. If `FALSE`, only the final configuration is returned.

Value

An object of class `nrMCmix`, containing the MCMC output, including:

log_p Log-posterior values across MCMC iterations.

n_pred_calls Number of predictive density evaluations at each iteration.

entropy Entropy of the allocation configuration.

K Number of occupied components at each iteration for nonparametric mixtures.

hyperpars Sampled hyperparameters.

acceptance_rates Acceptance rates of the MCMC moves, when available.

max_excursions Maximum number of excursions per scan, when available.

c Sampled allocation configurations, or the final configuration if `save_configurations = FALSE`.

atoms Sampled component parameters, when `save_parameters = TRUE`.

y The input data.

Examples

```
#specify the model
model <- specify_model("Gaussian", sigma2 = 1, lambda0 = 0.1)

#generate some data
set.seed(123)
y <- generate_data(n = 100, K = 7, model = model)$y

#run a non-reversible split and merge MCMC for a DP mixture
fit <- nrMCmix(y, K = 0, model = model,
 sampler = specify_sampler(SAM = TRUE))
```

`plot.nrMCmix` *Plot diagnostics for an* `nrMCmix` *model*

Description

Produces graphical summaries and diagnostics from a fitted `nrMCmix` object.

Usage

```
## S3 method for class 'nrMCmix'
plot(
  x,
  which_plot = 1L,
  line_col1 = NULL,
  line_col2 = NULL,
  x1_grid = NULL,
  x2_grid = NULL,
  idx1 = 1L,
  idx2 = NULL,
  B = 100L,
  l_grid = 101,
  n_breaks = NULL,
  verbose = TRUE,
  loss = salso::VI(),
  method = "complete",
  par_name = "log_p",
  lab_name = NULL,
  which_coordinate = 1L,
  f = function(x) x,
  ggplot2 = TRUE,
  ...
)
```

Arguments

x	An object of class nrMCmix.
which_plot	Integer specifying the type of plot to produce: 1 for data fit, 2 for dispersion, 3 for the posterior similarity matrix, 4 for a parameter trace plot, and 5 for a parameter density plot.
line_col1	A numeric or character scalar specifying the color of the posterior mean density when which_plot = 1, or the color of the main line in the density plot when which_plot = 5.
line_col2	A numeric or character scalar specifying the color of the area surrounding the main line when which_plot = 1 or which_plot = 5.
x1_grid	Optional numeric vector defining the grid used to evaluate the posterior density for the first variable. If NULL, a grid spanning the observed range of the corresponding variable is generated.
x2_grid	Optional numeric vector defining the grid used to evaluate the posterior density for the second variable in a bivariate Gaussian plot. If NULL, a grid spanning the observed range of the corresponding variable is generated.
idx1	Integer specifying the index of the first variable to plot. Defaults to 1.
idx2	Optional integer specifying the index of the second variable for bivariate plots.
B	Integer specifying the number of posterior atom draws used to approximate the posterior density. Defaults to 100.

`l_grid` Integer specifying the number of points in the automatically generated density grid.

`n_breaks` Optional integer specifying the number of histogram breaks for the univariate data-fit plot. If `NULL`, the number of breaks is chosen as `sqrt(n)`, where n is the number of observations.

`verbose` Logical indicating whether progress messages should be printed. Defaults to `TRUE`.

`loss` Loss function passed to `salso::salso()` when estimating the partition for the dispersion plot. Defaults to `salso::VI()`.

`method` Character string indicating the agglomeration method to be used to order the rows and columns of the posterior similarity matrix when `which_plot = 3`.

`par_name` Character string giving the name of the parameter to plot for `which_plot = 4` or `which_plot = 5`. Defaults to `"w"`.

`lab_name` Character string giving the label to be assigned at the parameter to plot when `which_plot = 4` or `which_plot = 5`.

`which_coordinate` Integer vector specifying the coordinate of the parameter to plot. A single integer selects an element of a vector parameter, while a vector of length two selects an element of a matrix parameter. Defaults to `1`.

`f` An R function that applies the function to the parameter of integer before returning it and/or selecting the desired dimension.

`ggplot2` A boolean value indicating whether the plot should utilize the grammar of graphics features. Default value is set to `TRUE`.

`...` additional arguments to customize plots.

Details

Five types of plots are available:

1. **Data fit**: for Gaussian, multivariate Gaussian, and Poisson kernels, displays the observed data together with posterior density draws and their pointwise mean. For multivariate Gaussian models, bivariate density plots can also be produced.
2. **Dispersion plot**: displays the observations coloured according to the estimated partition obtained using `salso::salso()`. For multivariate Gaussian models, bivariate dispersion plots can also be produced.
3. **Posterior similarity matrix**: displays the posterior similarity matrix computed from the sampled configurations.
4. **Trace plot**: displays the trace plot of a parameter (or function of parameters) across MCMC iterations, extracted using function `extract_params`.
5. **Density plot**: works as the previous one but displays the MCMC reconstruction of the posterior density.

For `which_plot = 1`, the available plots depend on the kernel. For multivariate Gaussian models, specifying `idx2` produces a bivariate density contour plot. Otherwise, a univariate histogram is overlaid with posterior density draws and their pointwise posterior mean. The posterior draws

of the density are obtained using the functions posterior_density1 for the univariate case and posterior_density2 for the bivariate case.

For which_plot = 2, the function estimates a representative partition using salso::salso() and colours observations according to their estimated cluster assignment. This plot requires multiple configuration draws to be available in X$c.

For which_plot = 3, the posterior similarity matrix is computed from X$c using function compute_similarity_matrix and it is displayed as a heatmap with the rows and columns ordered according to the output of hclust.

For which_plot = 4 and which_plot = 5, the requested parameter is extracted from the nrMCmix object using function extract_params.

Value

The function is called for its graphical side effect and does not return a value of interest.

Examples

```
#simulate multivariate gaussian data
set.seed(123)
y <- generate_data(n = 300, D = 2, K = 3,
                   model = specify_model("MGaussian", lambda0 = 0.9,
                                          dense = TRUE, alpha0 = 2.1))$y
plot(y)
#fit the model using nrMCmix
fit <- nrMCmix(y,model = specify_model("MGaussian", dense = TRUE,
                                        alpha0 = 2.01))

#model fit
{
  par(mfrow = c(2,2))

  #univariate density plot
  plot(fit, idx1 = 1, gg = FALSE)

  #bivariate density plots
  plot(fit, idx1 = 1, idx2 = 2, gg = FALSE)
  plot(fit, idx1 = 2, idx2 = 1, gg = FALSE)

  #univariate density plot
  plot(fit, idx1 = 2, gg = FALSE)
  par(mfrow = c(1,1))
}

#clustering
plot(fit, 2, idx1 = 1, idx2 = 2)

#posterior uncertainty using the similarity matrix
plot(fit,3)

#trace plot of the log posterior
plot(fit,4,par_name = "log_p")
```

```
#density plot of the numeber of cluster
plot(fit,5,par_name = "K")
```

---

posterior_density1 *Evaluate the posterior predictive density*

Description

Computes the posterior predictive density at a set of evaluation points for a single variable. The density is obtained by averaging over the mixture components and, for nonparametric models, the remaining prior probability mass.

Usage

```
posterior_density1(draw, hyperpars, xj, j = 0L)
```

Arguments

draw	List containing a posterior draw of the mixture parameters.
hyperpars	List containing the model hyperparameters.
xj	Numeric vector of points at which the density is evaluated.
j	Index of the variable to evaluate for multivariate Gaussian models. Defaults to 0 (C++ indexing).

Value

A numeric vector containing the posterior predictive density evaluated at each element of `xj`.

Examples

```
#specify the model
model <- specify_model("MGaussian", sigma2 = 1, lambda0 = 0.1)

#generate some data
set.seed(123)
y <- generate_data(n = 100, K = 7, model = model)$y

#run a non-reversible split and merge MCMC for a DP mixture
fit <- nrMCmix(y, K = 0, model = model,
 sampler = specify_sampler(SAM = TRUE))

#compute the density for the first draw
dens <- posterior_density1(fit$atoms[[1]],fit$hyperpars,seq(-5,5,l=101))
```

posterior_density2 *Evaluate the bivariate posterior predictive density*

Description

Computes the posterior predictive density on a two-dimensional grid of evaluation points for a pair of variables under the Gaussian mixture model. For nonparametric models, the contribution of the remaining prior probability mass is also included.

Usage

```
posterior_density2(draw, hyperpars, xj, xl, j = 0L, l = 1L)
```

Arguments

draw	List containing a posterior draw of the mixture parameters.
hyperpars	List containing the model hyperparameters.
xj	Numeric vector of evaluation points for the first variable.
xl	Numeric vector of evaluation points for the second variable.
j	Index of the first variable. Defaults to 0 (C++ indexing).
l	Index of the second variable. Defaults to 1 (C++ indexing).

Value

A numeric matrix containing the posterior predictive density on the grid defined by `xj` and `xl`.

Examples

```
#specify the model
model <- specify_model("MGaussian", sigma2 = 1, lambda0 = 0.1)

#generate some data
set.seed(123)
y <- generate_data(n = 100, K = 7, model = model)$y

#run a non-reversible split and merge MCMC for a DP mixture
fit <- nrMCmix(y, K = 0, model = model,
 sampler = specify_sampler(SAM = TRUE))

#compute the density for the first draw
dens <- posterior_density2(fit$atoms[[1]],fit$hyperpars,
                           seq(-5,5,l=101), seq(-4,4,l = 101),
                           0,1)
```

print.nrMCmix *Print an* nrMCmix *model fit*

Description

Prints the model specifications stored in an nrMCmix object.

Usage

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

Arguments

x	An object returned by the nrMCmix fitting procedure.
...	Additional values to pass. These currently do nothing.

Value

Invisibly returns NULL. Called primarily for its printing side effect.

Examples

```
#simulate data
set.seed(123)
y <- generate_data(n = 100, K = 3,
                   model = specify_model("Poisson",
                                         alpha0 = 10,
                                         beta0 = 3))$y[,1]

#fit the model
fit <- nrMCmix(y,K=3,model = specify_model("Poisson"))

#check the specifics
fit
```

print.summary.nrMCmix *Print an* nrMCmix *model summary*

Description

Prints the posterior summaries and MCMC diagnostics contained in an nrMCmix summary object.

Usage

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

Arguments

x	An object returned by [summary.nrMCmix()].
...	Additional values to pass. These currently do nothing.

Value

Invisibly returns NULL. Called primarily for its printing side effect.

---

priorK — *Prior distribution of the number of clusters*

Description

Computes the prior probability distribution of the number of clusters induced by a two-parameter nonparametric prior.

Usage

```
priorK(alpha = 1, delta = 0, n = 100L)
```

Arguments

alpha	Positive concentration parameter.
delta	Discount parameter satisfying.
n	Positive integer giving the number of observations. The returned vector contains the prior probabilities for $K = 1,..., n$.

Value

A numeric vector of length n, where the K-th element is the prior probability that the number of clusters is K.

Examples

```
priorK(alpha = 1, delta = 0, n = 100)
```

specify_model *Specify the model and prior distributions*

Description

Constructs a list specifying the kernel, clustering prior, and kernel-specific prior distributions used by the model.

Usage

```
specify_model(
  kernel = "Gaussian",
  alpha = NULL,
  delta = 0,
  hyperpar_alpha = FALSE,
  hyperpar_delta = FALSE,
  a0 = 0.01,
  b0 = 15,
  lwr_bound = 0,
  upr_bound = 1,
  mu0 = NULL,
  lambda0 = 0.01,
  sigma2 = NULL,
  alpha0 = 0.01,
  beta0 = 0.01,
  dense = FALSE,
  B0 = NULL,
  hyperpar_baseline = FALSE,
  m0 = NULL,
  V0 = 10,
  L1 = 0.1,
  L2 = 0.1,
  A1 = 0.1,
  A2 = 0.1,
  B1 = 0.1,
  B2 = 0.1,
  nu0 = 0.1,
  Psi0 = NULL
)
```

Arguments

`kernel`	Character string specifying the likelihood kernel. One of `"Gaussian"`, `"MGaussian"`, `"Poisson"`, `"Binomial"`, or `"MBernoulli"`.
`alpha`	Optional concentration parameter of the nonparametric prior.
`delta`	Discount parameter of the Pitman–Yor process.

hyperpar_alpha	Logical; if TRUE, a Gamma hyperprior is specified for the concentration parameter.
hyperpar_delta	Logical; if TRUE, a bounded prior is specified for the discount parameter.
a0, b0	Shape and rate parameters of the Gamma hyperprior for the concentration parameter.
lwr_bound, upr_bound	Lower and upper bounds for the discount parameter.
mu0	Prior mean parameter for Gaussian kernels.
lambda0	Prior precision parameter for Gaussian kernels.
sigma2	Optional known variance or covariance matrix. If NULL, the variance is estimated.
alpha0, beta0	Shape and rate parameters for Gamma or Beta priors, depending on the kernel.
dense	Logical; for "MGaussian", specifies whether an unstructured covariance matrix is used.
B0	Optional scale matrix for the inverse-Wishart prior used with a dense multivariate Gaussian kernel.
hyperpar_baseline	Logical; if TRUE, hyperpriors are specified for the baseline prior parameters.
m0, V0	Mean and variance parameters for the hyperprior on mu0.
L1, L2	Shape and rate parameters for the Gamma hyperprior on lambda0.
A1, A2	Shape and rate parameters for the Gamma hyperprior on alpha0.
B1, B2	Shape and rate parameters for the Gamma hyperprior on beta0.
nu0	Degrees of freedom parameter for the inverse-Wishart hyperprior.
Psi0	Optional scale or precision matrix for the inverse-Wishart hyperprior.

Value

A list containing the model specification and the corresponding prior and hyperprior parameters.

Examples

```
model <- specify_model(kernel = "Gaussian")

model <- specify_model(
  kernel = "MGaussian",
  dense = FALSE,
  hyperpar_alpha = TRUE
)
```

specify_sampler *Specify the sampling algorithm*

Description

Constructs a list specifying the options and tuning parameters of the sampling algorithm.

Usage

```
specify_sampler(
  reversible = FALSE,
  informed = FALSE,
  gibbs = FALSE,
  SAM = FALSE,
  NUSAMS = FALSE,
  xi = 0.5,
  s = 0,
  m = 20L,
  g = 0L,
  n_restricted_steps = 0L,
  thin_SAM = 1L,
  NU_weights = NULL,
  return_weights = FALSE,
  slice_max_iter = 10L,
  scale = 0.5
)
```

Arguments

reversible	Logical; if TRUE, use the reversible sampler for the local updates of the data allocation into groups. Defaults to FALSE.
informed	Logical; if TRUE, use informed proposals for the local updates of the data allocation into groups, unless gibbs = TRUE.
gibbs	Logical; if TRUE, use the marginal Gibbs sampler for the local updates of the data allocation into groups.
SAM	Logical; if TRUE, add split-and-merge global moves to the algorithm.
NUSAMS	Logical; if TRUE, use non-uniform weights within the split-and-merge algorithm.
xi	Tuning parameter for the non-reversible sampler, it regulates the refreshment rate of the directional variable by reshuffling them at each iteration with probability xi$/n$.
s	Parameter that, when greater than 0, employs a modified non-reversible sampler that sample the number of consecutive iterations along the same direction from a geometric distribution of parameter $s/(n_k + n_{k'})$ where n_k denotes the size of the k-th group. This version is available only for the finite case (i.e., when K > 0).

m	Upper bound on the random neighborhood's size used to construct the informed proposal.
g	Type of balancing function used by the informed proposal. Setting g=0 corresponds to the square root function: $g(t) = \sqrt{t}$, g=1 corresponds to the Metropolis function: $g(t) = min{1,t}$, while g=2 to the Barker: $g(t) = t /(1+t)$
n_restricted_steps	Number of restricted Gibbs steps used in split-and-merge global proposals.
thin_SAM	Number of subsequent global proposals after a sequence of local ones.
NU_weights	Optional matrix of non-uniform weights used by the split-and-merge algorithm.
return_weights	Logical; if TRUE, return the non-uniform weights generated by the sampler.
slice_max_iter	Maximum number of iterations allowed by the slice sampler.
scale	Scale parameter of the slice sampler.

Value

A list containing the selected sampling options and the corresponding tuning parameters.

Examples

```
sampler <- specify_sampler()

sampler <- specify_sampler(
  reversible = FALSE,
  informed = TRUE,
  m = 20
)

sampler <- specify_sampler(
  SAM = TRUE,
  NUSAMS = TRUE
)
```

---

summary.nrMCmix *Summarize an nrMCmix model fit*

Description

Computes posterior summary statistics and MCMC diagnostics for an nrMCmix model fit. Summaries are provided for the log-posterior, entropy, and, when available, the number of mixture components. Acceptance rates and excursion statistics are also reported when available.

Usage

```
## S3 method for class 'nrMCmix'
summary(object, ...)
```

Arguments

`object`	An object returned by the `nrMCmix` fitting procedure.
`...`	Additional values to pass. These currently do nothing.

Value

A list containing:

posterior_summaries A data frame containing the minimum, first quartile, median, mean, third quartile, maximum, and effective sample size (ESS) for each monitored quantity.

split_rate Mean split-move acceptance rate, when available.

merge_rate Mean merge-move acceptance rate, when available.

local_acceptance_rate Mean local-move acceptance rate, when available.

excursions Median maximum excursion, when available.

Examples

```
#simulate data
set.seed(123)
y <- generate_data(n = 100, K = 3,
                   model = specify_model("Poisson",
                                         alpha0 = 10,
                                         beta0 = 3))$y[,1]

#fit the model
fit <- nrMCmix(y,K=3,model = specify_model("Poisson"))

#check the summary
summary(fit)
```

Index

∗ **package**
    nrMCmix-package, 2

compute_similarity_matrix, 5, *12*

extract_params, 6, *11*, *12*

generate_data, 7

hclust, *12*

nrMCmix, *6*, 8
nrMCmix-package, 2

plot.nrMCmix, 9
posterior_density1, *12*, 13
posterior_density2, 14
print.nrMCmix, 15
print.summary.nrMCmix, 15
priorK, 16

specify_model, *7*, *8*, 17
specify_sampler, *8*, 19
summary.nrMCmix, 20