Package: MultiModalR (via r-universe)

October 5, 2026

Title Bayesian Mixture Modeling for Stratified and Hierarchical Multimodal Data

Version 1.1.0

Date 2026-08-29

Description Fast Bayesian probability estimation for multimodal continuous data using speed-optimized Markov chain Monte Carlo (MCMC) implementation. The package provides two complementary approaches: (1) stratified mixture modeling with independent fits per category (fuss_PARALLEL_mcmc), and (2) a unified hierarchical mixture model with category-specific parameters and shared number of components (fuss_COVARIATE_mcmc). Both approaches detect subpopulations, estimate mixture components, and assign observations to subgroups with probability estimates. The methods are described in Dioszegi, G. et al. (2026) ``Automatic Bayesian Mixture Modeling for Multimodal Categorical Data via Integrated Mode Detection and Metropolis-Hastings-within-Gibbs Sampling" (submitted).

License MIT + file LICENSE

URL https://github.com/DijoG/MultiModalR

BugReports https://github.com/DijoG/MultiModalR/issues

Depends R (>= 3.5.0)

Imports Rcpp (>= 1.0.10), dplyr, purrr, readr, ggplot2, furrr, future, truncnorm, rlang

Suggests testthat (>= 3.0.0), knitr, rmarkdown, multimode, tictoc, tidyr

LinkingTo Rcpp, RcppArmadillo

SystemRequirements C++17

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation yes

LazyData true

Config/pak/sysreqs libx11-dev

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

Date/Publication 2026-09-05 16:56:06 UTC

RemoteUrl https://github.com/dijog/multimodalr

RemoteRef HEAD

RemoteSha 434985d5d3640cf41dcd1cd87e9eee30555c439a

Contents

check_convergence 2
check_PACKS 3
create_MM_output 3
create_multimodal_dummy 4
effective_sample_size 5
fuss_COVARIATE_mcmc 5
fuss_PARALLEL_mcmc 6
gelman_rubin 7
get_MODES_enhanced 8
group_MODES_enhanced 8
MM_MH 9
MM_MH_dirichlet 10
multimodal_dummy 11
plot_VALIDATION 12

Index **14**

`check_convergence` *Check MCMC convergence*

Description

Check MCMC convergence

Usage

```
check_convergence(object, chains = NULL)
```

Arguments

`object`	A `fuss_COVARIATE_mcmc` object
`chains`	Optional list of multiple chains for Gelman-Rubin

Value

Convergence diagnostics

check_PACKS *Check if required packages are available*

Description

Check if required packages are available

Usage

```
check_PACKS()
```

Value

Invisibly returns TRUE if all packages are available

create_MM_output *Create output data frame*

Description

Converts MCMC results to exact CSV format

Usage

```
create_MM_output(
  mcmc_result,
  y_original = NULL,
  group_original = NULL,
  main_class = "",
  max_groups = 5
)
```

Arguments

mcmc_result	Output from MM_MH() or MM_MH_dirichlet()
y_original	Original y values (if different from mcmc_result$y)
group_original	Original group assignments (optional)
main_class	Category/class name
max_groups	Maximum number of groups for output columns

Value

Data frame in CSV format

`create_multimodal_dummy` *Create multimodal dummy dataset*

Description

Generates the same dummy dataset used in the package. This is useful if users want to generate similar data with different parameters.

Usage

```
create_multimodal_dummy(
  seed = 5,
  n_categories = 9,
  n_per_group = 25,
  n_subgroups = 3
)
```

Arguments

`seed`	Random seed for reproducibility (default: 5)
`n_categories`	Number of categories (default: 9)
`n_per_group`	Number of observations per subgroup per category (default: 25)
`n_subgroups`	Number of subgroups per category (default: 3)

Value

A data frame with multimodal data

Examples

```
# Generate the default dataset
df <- create_multimodal_dummy()

# Generate with different parameters
df2 <- create_multimodal_dummy(seed = 12, n_categories = 6)
```

effective_sample_size *Compute Effective Sample Size (ESS) for MCMC chains*

Description

Compute Effective Sample Size (ESS) for MCMC chains

Usage

```
effective_sample_size(x)
```

Arguments

x	Numeric vector or matrix of MCMC samples

Value

Effective sample size

fuss_COVARIATE_mcmc *Unified Bayesian Mixture Model with Category-Specific Parameters*

Description

Fits a Bayesian Gaussian mixture model where each category can have a different number of components (K_c), while sharing information across categories for means and variances.

Usage

```
fuss_COVARIATE_mcmc(
  data,
  varY,
  varCLASS,
  varID = NULL,
  K = NULL,
  n_iter = 10000,
  burnin = 2000,
  proposal_sd = 0.15,
  adaptive = TRUE,
  alpha0 = 3,
  beta0 = 2,
  alpha_dirichlet = 5,
  method = "sj-dpi",
  sj_adjust = 0.5,
  within = 1,
  seed = 123
)
```

Arguments

data	Data frame containing the variables.
varY	Name of the continuous response variable.
varCLASS	Name of the categorical grouping variable.
varID	Optional name of the ID variable.
K	Optional vector of K per category. If NULL, auto-detected.
n_iter	Number of MCMC iterations (default: 10000).
burnin	Number of burn-in iterations (default: 2000).
proposal_sd	Proposal standard deviation (default: 0.15).
adaptive	Logical; adapt proposal variance (default: TRUE).
alpha0	Shape parameter for Inverse Gamma prior (default: 3.0).
beta0	Scale parameter for Inverse Gamma prior (default: 2.0).
alpha_dirichlet	Concentration parameter (default: 5.0).
method	Bandwidth selection method (default: "sj-dpi").
sj_adjust	Adjustment factor (default: 0.5).
within	Merging radius (default: 1.0).
seed	Random seed (default: 123).

Value

A list with components.

---

fuss_PARALLEL_mcmc *Parallel Bayesian mixture modeling using Markov Chain Monte Carlo (MCMC)*

Description

Performs multimodal probability assignment using either: 1. Metropolis-Hastings-within-partial-Gibbs 2. Dirichlet-Multinomial

Usage

```
fuss_PARALLEL_mcmc(
  data,
  varCLASS,
  varY,
  varID,
  method = "sj-dpi",
  within = 1,
  maxNGROUP = 5,
```

```
  out_dir = NULL,
  n_workers = 3,
  n_iter = NULL,
  burnin = NULL,
  proposal_sd = 0.15,
  sj_adjust = 0.5,
  mcmc_method = "metropolis",
  dirichlet_alpha = 2
)
```

Arguments

data	Input data frame
varCLASS	Character, category variable name (required)
varY	Character, value variable name (required)
varID	Character, ID variable name (required)
method	Density estimator method ("sj-dpi", "bcv", "ucv", "nrd") (default: "sj-dpi")
within	Range parameter for grouping modes (default: 1.0)
maxNGROUP	Maximum number of groups (default: 5)
out_dir	Output directory for CSV files (if NULL, returns combined data frame)
n_workers	Number of parallel workers (default: 3)
n_iter	Number of MCMC iterations (default: 6000 for metropolis, 3000 for dirichlet)
burnin	Burn-in period (default: 2000 for metropolis, 1000 for dirichlet)
proposal_sd	Proposal standard deviation for component means (default: 0.15)
sj_adjust	Adjustment factor for bandwidth methods (default: 0.5)
mcmc_method	"metropolis" or "dirichlet" (default: "metropolis")
dirichlet_alpha	Dirichlet concentration parameter (default: 2.0)

Value

Data frame (ALWAYS returns combined data frame, even when writing CSV)

---

gelman_rubin *Compute Gelman-Rubin R_hat for MCMC chains*

Description

Compute Gelman-Rubin R_hat for MCMC chains

Usage

```
gelman_rubin(chains)
```

Arguments

chains	List of either: - matrices (each row = parameter, each column = iteration), or - fuss_COVARIATE_mcmc result objects

Value

Vector of R_hat values (one per parameter)

---

get_MODES_enhanced *Density height-aware mode detection*

Description

Returns mode estimates from FOUR different bandwidth methods. Each method may detect different numbers and locations of modes.

Usage

```
get_MODES_enhanced(y, adjust = 1, threshold = 1)
```

Arguments

y	Numeric vector
adjust	Bandwidth adjustment factor (affects "SJ", "nrd", "bcv" methods)
threshold	Relative threshold for significant peaks

Value

List with mode estimates from multiple methods including density heights

---

group_MODES_enhanced *Density height-aware mode grouping*

Description

Density height-aware mode grouping

Usage

```
group_MODES_enhanced(df, within = 0.1)
```

Arguments

df	data frame containing 'Est_Mode' and 'Density_Height' columns
within	numeric, range for grouping modes (default: 0.1)

Value

data frame with grouped modes

`MM_MH` *Fast MCMC for mixture models (Metropolis-Hastings-within-partial-Gibbs)*

Description

Fast MCMC for mixture models (Metropolis-Hastings-within-partial-Gibbs)

Usage

```
MM_MH(
  y,
  grp,
  prior_means = NULL,
  ids,
  n_iter = 1000,
  burnin = 500,
  proposal_sd = 0.15,
  seed = NULL
)
```

Arguments

`y`	Numeric vector of data
`grp`	Number of mixture components
`prior_means`	Prior means for components (optional)
`ids`	Vector of IDs for validation (required)
`n_iter`	Number of MCMC iterations (default: 1000)
`burnin`	Burn-in period (default: 500)
`proposal_sd`	Proposal standard deviation for component means (default: 0.15)
`seed`	Random seed

Value

List with MCMC results

Examples

```
# Simulate data
set.seed(123)
y <- c(rnorm(50, 0, 1), rnorm(50, 5, 1))
ids <- paste0("ID", 1:100)

# Run MCMC
result <- MM_MH(y, grp = 2, ids = ids, n_iter = 100, burnin = 50)
print(head(result$assigned_group))
```

`MM_MH_dirichlet` *Dirichlet MCMC (identical interface to MM_MH)*

Description

Dirichlet MCMC (identical interface to MM_MH)

Usage

```
MM_MH_dirichlet(
  y,
  grp,
  prior_means = NULL,
  ids,
  n_iter = 5000,
  burnin = 2000,
  proposal_sd = 0.15,
  dirichlet_alpha = 2,
  seed = NULL
)
```

Arguments

`y`	Numeric vector of data
`grp`	Number of mixture components
`prior_means`	Prior means for components
`ids`	Vector of IDs for validation
`n_iter`	Number of MCMC iterations
`burnin`	Burn-in period
`proposal_sd`	Proposal standard deviation
`dirichlet_alpha`	Dirichlet concentration parameter
`seed`	Random seed

Value

List with MCMC results (SAME FORMAT as MM_MH)

Examples

```
# Simulate data
set.seed(123)
y <- c(rnorm(50, 0, 1), rnorm(50, 5, 1))
ids <- paste0("ID", 1:100)

# Run Dirichlet MCMC
result <- MM_MH_dirichlet(y, grp = 2, ids = ids,
                          n_iter = 100, burnin = 50)
print(head(result$assigned_group))
```

---

multimodal_dummy *Multimodal Dummy Dataset*

Description

A simulated dataset containing 9 categories each with 3 distinct subpopulations following truncated normal distributions. Ideal for testing multimodal mixture modeling.

A simulated dataset containing 9 categories (AA, BB, ..., II) each with 3 distinct subpopulations (Group 1, Group 2, Group 3) following truncated normal distributions.

Usage

```
data(multimodal_dummy)

multimodal_dummy
```

Format

A data frame with 675 rows and 4 columns

A data frame with 675 rows and 4 variables:

Category Factor with 9 levels: AA, BB, CC, DD, EE, FF, GG, HH, II

Subpopulation Factor with 3 levels: Group 1, Group 2, Group 3

Value Numeric values between 5 and 10

ID Unique identifier from 1 to 675

Details

This dataset is useful for demonstrating the capabilities of MultiModalR package. Each category contains three distinct subpopulations with overlapping but separable distributions, making it ideal for testing multimodal mixture modeling algorithms.

Source

Generated with `set.seed(5)` using `truncnorm::rtruncnorm()`. See `create_multimodal_dummy` for the generating function.

Examples

```
# Load the dataset
data(multimodal_dummy)

# View structure
str(multimodal_dummy)

# Summary statistics
summary(multimodal_dummy)

# Plot data
library(ggplot2)
ggplot(multimodal_dummy, aes(x = Value, fill = Subpopulation)) +
  geom_density(alpha = 0.5, color = NA) +
  facet_wrap(~Category) +
  theme_dark()

# Use with MultiModalR

library(MultiModalR)
results <- fuss_PARALLEL_mcmc(
  data = multimodal_dummy,
  varCLASS = "Category",
  varY = "Value",
  varID = "ID",
  n_workers = 3
)
```

---

plot_VALIDATION *Plot validation of subgroup assignments (handles both balanced and imbalanced data)*

Description

Plot validation of subgroup assignments (handles both balanced and imbalanced data)

Usage

```
plot_VALIDATION(
  csv_dir,
  observed_df,
  subpop_col = "Subpopulation",
  value_col = "Value",
```

```
  id_col = "ID",
  pattern = "^df_"
)
```

Arguments

csv_dir	Directory containing CSV files from create_MM_output
observed_df	Original data frame with true subgroups
subpop_col	Character, name of the true subgroup column in observed_df (default: "Subpopulation")
value_col	Character, name of the value column in observed_df (default: "Value")
id_col	Character, name of the ID column in observed_df (default: "ID")
pattern	Pattern to match CSV files (default: "^df_")

Value

ggplot object showing validation results

Index

∗ **datasets**
multimodal_dummy, 11

check_convergence, 2
check_PACKS, 3
create_MM_output, 3
create_multimodal_dummy, 4, *12*

effective_sample_size, 5

fuss_COVARIATE_mcmc, 5
fuss_PARALLEL_mcmc, 6

gelman_rubin, 7
get_MODES_enhanced, 8
group_MODES_enhanced, 8

MM_MH, 9
MM_MH_dirichlet, 10
multimodal_dummy, 11

plot_VALIDATION, 12