Package: MultiModalR (via r-universe)

July 1, 2026

Title Fast Bayesian Probability Estimation for Multimodal Categorical Data

Version 1.0.0

Date 2026-06-18

Description Fast Bayesian probability estimation for multimodal categorical data using speed-optimized MCMC implementation (Metropolis-Hastings-within-partial-Gibbs). The package provides efficient algorithms for detecting subpopulations, estimating mixture components, and assigning observations to subgroups with probability estimates.

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 2026-06-23 12:43:01 UTC

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

RemoteRef HEAD

RemoteSha d235608b9f70128dab5382cce870baa9b7892b09

Contents

`check_PACKS` *Check and install required packages*

Description

Check and install required packages

Usage

```
check_PACKS()
```

Value

Installs missing packages

`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)
```

`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, smaller -> more modes, higher -> fewer modes)
`mcmc_method`	"metropolis" or "dirichlet"(default: "metropolis")
`dirichlet_alpha`	Dirichlet concentration parameter (default: 2.0)

Value

Data frame (if out_dir is NULL) or writes CSV files

`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

`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)

`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 rtruncnorm package

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
## Not run:
library(MultiModalR)
results <- fuss_PARALLEL_mcmc(
  data = multimodal_dummy,
  varCLASS = "Category",
  varY = "Value",
  varID = "ID",
  n_workers = 3
)

## End(Not run)
```

`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