Package: simBKMRdata (via r-universe)

May 24, 2026

Title Helper Functions for Bayesian Kernel Machine Regression

Version 0.2.1

Description Provides a suite of helper functions to support Bayesian Kernel Machine Regression (BKMR) analyses in environmental health research. It enables the simulation of realistic multivariate exposure data using Multivariate Skewed Gamma distributions, estimation of distributional parameters by subgroup, and application of adaptive, data-driven thresholds for feature selection via Posterior Inclusion Probabilities (PIPs). It is especially suited for handling skewed exposure data and enhancing the interpretability of BKMR results through principled variable selection. The methodology is shown in Hasan et. al. (2025) <doi:10.1101/2025.04.14.25325822>.

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

License GPL (>= 3)

Imports MASS, stats

Suggests bkmr, fields, gt, quarto, testthat (>= 3.0.0), tidyverse

VignetteBuilder quarto

Depends R (>= 3.5)

LazyData true

Config/testthat/edition 3

Repository https://khasa006.r-universe.dev

Date/Publication 2025-05-18 04:29:24 UTC

RemoteUrl https://github.com/khasa006/simbkmrdata

RemoteRef HEAD

RemoteSha a94acbec92f9cbce06ac7497342c0d468a3dc636

Contents

`.skewness` *Helper function to calculate skewness for a vector*

Description

Helper function to calculate skewness for a vector

Usage

```
.skewness(x, mean_vec, sampSD, N)
```

Arguments

`x`	A numeric vector of data
`mean_vec`	The mean of the data
`sampSD`	The standard deviation of the data
`N`	The sample size

Value

The skewness value

`calculate_pip_threshold` *Calculate PIP Threshold*

Description

Given a response vector (or statistics from this vector), calculate a PIP threshold that should preserve close to a nominal 5% test size for Bayesian Kernel Machine Regression (BKMR) feature selection.

Usage

```
calculate_pip_threshold(
  y,
  absCV,
  sampSize,
 coeffs_ls = list(A = 0, K = 1, C = 1.3046, betaAbsCV = 0.59867, betaSampSize = 0.43565),
  na.rm = TRUE
)
```

Arguments

`y`	a response vector for BKMR
`absCV`	If `y` is not supplied, the absolute value of the coefficient of variation of the response
`sampSize`	If `y` is not supplied, the number of observations included in the response
`coeffs_ls`	A list of Richard's Curve parameters. See Details.
`na.rm`	Remove missing values from `y`? Defaults to `TRUE`

Details

CalculatePipThreshold function is designed to model the relationship between PIP(q95), coefficient of variation (CV), and sample size using a form of four-parameter logistic regression (Richard Curve). This function employs the `nls` function from the R `stats` package, utilizing the Levenberg-Marquardt algorithm for optimization to ensure robust parameter estimation.

$$PIP(q_{95}) = A + \frac{K - A}{(C + \exp(-\beta_1 x_1))^{\beta_2 x_2}}$$

Where-

A: Fixed left asymptote (0);

K: Right asymptote;

C: Constant;

β_1, β_2: Midpoint shift parameters for CV and sample size;

x1: Log2-transformed |CV| (log2(|CV|));

x2: Log-transformed sample size (log10(Sample Size)).

The detailed explanation of how we calculated the values in `coeffs_ls` can be found in <.....>.

For more information on Richard's curve, see https://en.wikipedia.org/wiki/Generalised_logistic_function

Value

A single numeric value; the output of the Richard's Four-Parameter Logistic Regression curve with the coefficient values supplied in `coeffs_ls`.

Examples

```
calculate_pip_threshold(absCV = 7.5, sampSize = 300)
# should equal 0.6829892
```

`calculate_stats_gamma` *Calculate summary statistics and gamma parameters for each group*

Description

This function computes the sample size, gamma distribution parameters (shape and rate), and Spearman correlation matrix for each group, based on the grouping column.

Usage

```
calculate_stats_gamma(data_df, group_col, using = c("MoM", "gMLE"))
```

Arguments

`data_df`	A data frame containing the data to be processed.
`group_col`	A character string specifying the name of the column to group by.
`using`	which method will be used to estimate the multivariate Gamma shape and rate parameters. Defaults to `"MoM"` (method of moments, which was used in the author's paper), or `"gMLE"` (maximum likelihood estimates from the Generalized Gamma distribution without bias correction).

Value

A list of lists, where each inner list contains:

- sample size (n)
- sample mean vector (mean)
- gamma distribution parameters (shape, rate)
- Spearman correlation matrix (cor)

Examples

```
myData <- data.frame(
  GENDER = c('Male', 'Female', 'Male', 'Female', 'Male', 'Female'),
  VALUE1 = c(1.2, 2.3, 1.5, 2.7, 1.35, 2.5),
  VALUE2 = c(3.4, 4.5, 3.8, 4.2, 3.6, 4.35)
)
calculate_stats_gamma(myData, "GENDER")
```

---

calculate_stats_gaussian

Calculate summary statistics for each group

Description

This function computes the sample size, mean vector, standard deviation vector, Spearman correlation matrix, and skewness vector for each group, based on the grouping column.

Usage

```
calculate_stats_gaussian(data_df, group_col)
```

Arguments

`data_df`	A data frame containing the data to be processed.
`group_col`	A character string specifying the name of the column to group by.

Value

A list of lists, where each inner list contains the following parameter estimates for one group:

- sample size (`sampSize`)
- sample mean vector (`xBar`)
- sample standard deviation vector (`sampSD`)
- sample Spearman correlation matrix (`sampCorr_mat`)
- sample skewness (`sampSkew`)

Examples

```
myData <- data.frame(
  GENDER = c('Male', 'Female', 'Male', 'Female', 'Male', 'Female'),
  VALUE1 = c(1.2, 2.3, 1.5, 2.7, 1.35, 2.5),
  VALUE2 = c(3.4, 4.5, 3.8, 4.2, 3.6, 4.35)
)
calculate_stats_gaussian(myData, "GENDER")
```

`estimate_mv_moments` *Helper function to estimate moment vectors/matrices for observations within a group*

Description

Helper function to estimate moment vectors/matrices for observations within a group

Usage

```
estimate_mv_moments(x_df)
```

Arguments

`x_df`	A numeric data frame with observations from ONE group

Value

A list of statistics/moments (sample size, mean, standard deviation, correlation matrix, skewness) as vectors/matrices

Examples

```
myData <- data.frame(
  VALUE1 = c(2.3, 2.7, 2.5),
  VALUE2 = c(4.5, 4.2, 4.35)
)
estimate_mv_moments(myData)
```

`estimate_mv_shape_rate` *Helper function to estimate shape, rate, and correlation parameters for observations within a group*

Description

Helper function to estimate shape, rate, and correlation parameters for observations within a group

Usage

```
estimate_mv_shape_rate(x_df, using = c("MoM", "gMLE"))
```

Arguments

x_df	A numeric data frame with observations from ONE group
using	which method will be used to estimate the multivariate Gamma shape and rate parameters. Defaults to "MoM" (method of moments, which was used in the author's paper), or "gMLE" (maximum likelihood estimates from the Generalized Gamma distribution without bias correction).

Value

A list of estimated parameters for Multivariate Gamma distribution (sample size, sample mean, sample correlation matrix sampCorr_mat, sample shape vector alpha, sample rate vector beta)

Examples

```
myData <- data.frame(
  VALUE1 = c(2.3, 2.7, 5),
  VALUE2 = c(4.5, 4.2, 9)
)
estimate_mv_shape_rate(myData)
```

---

generate_mvGamma_data *Generate Multivariate Skewed Gamma Transformed Data*

Description

This function generates multivariate normal samples, transforms them into Z-scores, and then calls the qgamma() function to transform the values for each correlated variable to those from a Gamma distribution.

Usage

```
generate_mvGamma_data(sampSize, sampCorr_mat, shape_num, rate_num)
```

Arguments

sampSize	Number of samples to generate.
sampCorr_mat	A correlation matrix for the normal distribution.
shape_num	A numeric vector of shape parameters for the Gamma transformation.
rate_num	A numeric vector of rate parameters for the Gamma transformation. Second column: https://en.wikipedia.org/wiki/Gamma_distribution

Value

A data frame containing the transformed Gamma samples.

Examples

```
p <- 4
N <- 1000
shapeGamma_num <- c(0.5, 0.75, 1, 1.25)
rateGamma_num <- 1:4
cov_mat <- diag(p)
generate_mvGamma_data(N, cov_mat, shapeGamma_num, rateGamma_num)
```

---

metalExposChildren_df *Children's Environmental Heavy Metal Exposure Data*

Description

This dataset originates from a real-world cohort study led by Dr. Lucchini and collaborators, focusing on environmental exposures in children. It includes measurements of five metal concentrations—Cadmium, Mercury, Arsenic, Lead, and Manganese—alongside standardized intelligence quotient (IQ) scores, and some other socio-economic and demographic variables.

Usage

```
metalExposChildren_df
```

Format

who:

A data frame with 437 rows and 13 columns:

ID Subject's ID

age Subject's age in years

QI Intelligence quotient

Cadmium Cd urine concentration ng/ml

Mercury Hg urine concentration ng/ml

Arsenic As urine concentration ng/ml

Lead Pb blood concentration ng/ml

Manganese Mn hair concentration ng/g

Sex Student's gender; 1 = Male, 2 = Female

BMI Body Mass Index (kg/m2)

SES Social Economic Status; 1 = LOW, 2 = MEDIUM, 3 = HIGH

Distance_metres Distance to nearest industrial site (m)

SPM Raven's Standard Progressive Matrices

Source

Prof. Roberto Lucchini; see inst/scripts/data_metalExposChildren*.R for more details. Licensed under CC BY 4.0.

simulate_group_data *Simulate Group Data*

Description

This function generates data for each group by invoking the specified data generation function once per group. It binds the generated data together into a single data frame.

Usage

```
simulate_group_data(param_list, data_gen_fn, group_col_name)
```

Arguments

param_list	A list of named sublists, where each sublist contains the parameters for a group (mean, shape, rate, etc.). The list must be named with group names that match the groupings stated in group_col_name.
data_gen_fn	A function for data generation. Currently we can choose either generate_mvGamma_data or MASS::mvrnorm.
group_col_name	The name of the grouping/label column to be created in the final data frame.

Value

A data frame with the simulated data for all groups, including the grouping column.

Examples

```
# Example using MASS::mvrnorm for normal distribution
param_list <- list(
 Group1 = list(mean_vec = c(1, 2), sampCorr_mat = matrix(c(1, 0.5, 0.5, 1), 2, 2), sampSize = 100),
 Group2 = list(mean_vec = c(2, 3), sampCorr_mat = matrix(c(1, 0.3, 0.3, 1), 2, 2), sampSize = 150)
)
simulate_group_data(param_list, MASS::mvrnorm, "Group")

# Example using generate_mvGamma_data for Gamma distribution
param_list <- list(
  Group1 = list(sampCorr_mat = matrix(c(1, 0.5, 0.5, 1), 2, 2),
                shape_num = c(2, 2), rate_num = c(1, 1), sampSize = 100),
  Group2 = list(sampCorr_mat = matrix(c(1, 0.3, 0.3, 1), 2, 2),
                shape_num = c(2, 2), rate_num = c(1, 1), sampSize = 150)
)
simulate_group_data(param_list, generate_mvGamma_data, "Group")
```

`simulate_group_gamma` *Simulate Group Multivariate Data*

Description

This function generates data for each group from a Multivariate Gamma Distribution by invoking this distribution's random generator once per group. It binds the generated data together into a single data frame.

Usage

```
simulate_group_gamma(param_list, group_col_name)
```

Arguments

`param_list`	A list of named sublists, where each sublist contains the parameters for a group (sample size, mean, standard correlation matrix, shape, and rate parameter). The dimension of the parameters for each group must be the same.
`group_col_name`	The column name of the grouping/label column to be created in the final data frame. The values are taken from the names of the sublists of `param_list`. Defaults to "group". See the example below.

Value

A data frame with the simulated data for all groups, including the grouping column.

Examples

```
# Example using generate_mvGamma_data for MV Gamma distribution
param_list <- list(
 Male = list(
  sampSize = 100,
  sampCorr_mat = matrix(c(1, 0.5, 0.5, 1), 2, 2),  # Covariance matrix
  shape_num = c(2, 2),  # Shape parameters for Gamma distribution
  rate_num = c(1, 1)    # Rate parameters for Gamma distribution
 ),
 Female = list(
  sampSize = 150,
  sampCorr_mat = matrix(c(1, 0.3, 0.3, 1), 2, 2),
  shape_num = c(1, 4),
  rate_num = c(0.5, 2)
 )
)
simulate_group_gamma(param_list, "Sex")
```

simulate_group_gaussian *Simulate Group Multivariate Gaussian Data*

Description

This function generates data for each group from a Multivariate Gaussian (Normal) Distribution by invoking this distribution's random generator once per group. It binds the generated data together into a single data frame.

Usage

```
simulate_group_gaussian(param_list, group_col_name)
```

Arguments

param_list	A list of named sublists, where each sublist contains the parameters for a group (sample size, mean, standard deviation, and correlation matrix). The dimension of the parameters for each group must be the same.
group_col_name	The column name of the grouping/label column to be created in the final data frame. The values are taken from the names of the sublists of param_list. Defaults to "group". See the example below.

Value

A data frame with the simulated data for all groups, including the grouping column.

Examples

```
# Example using MASS::mvrnorm for normal distribution
param_list <- list(
  Male = list(
    sampSize = 50,
    mean_vec = c(1, 2),
    sampSD = c(2, 1),
    sampCorr_mat = matrix(c(1, 0.5, 0.5, 1), 2, 2)
  ),
  Female = list(
    sampSize = 100,
    mean_vec = c(2, 3),
    sampSD = c(1, 2),
    sampCorr_mat = matrix(c(1, 0.3, 0.3, 1), 2, 2)
  )
)
simulate_group_gaussian(param_list, "Sex")
```

`transformers` *Data Transformation Functions*

Description

This script provides transformation functions for data scaling and normalization.

Usage

```
trans_ratio(x, method = c("sd", "mad"))

trans_root(x, fracRoot = 0.5)

trans_log(x, base = 10, shift = 1)
```

Arguments

`x`	A numeric vector or column of a dataframe to be transformed.
`method`	Character string specifying the method: "sd" (standard deviation) or "mad" (median absolute deviation).
`fracRoot`	The fractional power to which each element in `x` should be raised. Defaults to 0.5 (square root).
`base`	The base of the logarithm. Defaults to 10.
`shift`	A numeric value added to `x` before applying the logarithm to avoid log(0). Defaults to 1.

Value

A numeric vector where values are divided by the chosen method's statistic.

A numeric vector of transformed values.

A numeric vector of transformed values.

Examples

```
trans_ratio(c(1, 2, 3, 4, 5), method = "sd")
trans_root(c(1, 4, 9, 16), fracRoot = 0.5)
trans_log(c(1, 10, 100, 1000), base = 10, shift = 1)
```

Index