Package ‘circlus’

February 19, 2025

Type Package

Title Clustering and Simulation of Spherical Cauchy and PKBD Models

Version 0.0.2

Description Provides tools for estimation and clustering of spherical data, seamlessly integrated with the 'flexmix' package. Includes the necessary M-step implementations for both Poisson Kernel-Based Distribution (PKBD) and spherical Cauchy distribution. Additionally, the package provides random number generators for PKBD and spherical Cauchy distribution. Methods are based on Golzy M., Markatou M. (2020) <doi:10.1080/10618600.2020.1740713>, Kato S., McCullagh P. (2020) <doi:10.3150/20-bej1222> and Sablica L., Hornik K., Leydold J. (2023) <doi:10.1214/23-ejs2149>.

License GPL-3

Encoding UTF-8

Imports Rcpp (>= 0.12.18), Tinflex (>= 1.8), flexmix, torch, methods

LinkingTo Rcpp, RcppArmadillo

URL `https://github.com/lsablica/circlus`

BugReports `https://github.com/lsablica/circlus/issues`

RoxygenNote 7.3.2

Depends R (>= 3.1.0)

LazyData true

NeedsCompilation yes

Author Lukas Sablica [aut, cre] (<`https://orcid.org/0000-0001-9166-4563`>),
Kurt Hornik [aut] (<`https://orcid.org/0000-0003-4198-9911`>),
Bettina Gruen [aut] (<`https://orcid.org/0000-0001-7265-4773`>),
Josef Leydold [aut] (<`https://orcid.org/0000-0002-9076-4893`>)

Maintainer Lukas Sablica <`lsablica@wu.ac.at`>

Repository CRAN

Date/Publication 2025-02-19 17:00:07 UTC

Contents

`Abstracts` *Abstracts of Papers Co-Authored by Friedrich Leisch*

Description

A dataset containing the abstracts of papers co-authored by Friedrich Leisch together with obtained metadata and embeddings generated using GMT and OpenAI embeddings models.

Usage

```
Abstracts
```

Format

A data frame with 129 rows and 282 variables.

`dpkb` *Density Function for PKB distribution*

Description

Calculates the density of the PKB distribution for given data points.

Usage

```
dpkb(y, mu, rho, log = FALSE)
```

Arguments

`y` A matrix or data frame where each row represents a data point on the unit hypersphere.

`mu` A vector or matrix representing the mean direction parameter(s). If a vector, it must be normalized (unit length) and is applied to all data points. If a matrix, it must have the same number of rows as `y`, and each row must be normalized.

`rho` A scalar or a vector representing the concentration parameter. If a vector, its length must match the number of rows in `y`. Each `rho[i]` is used to evaluate the density for `y[i, ]`. Must be between 0 (inclusive) and 1 (exclusive).

`log` Logical; if TRUE, the log-density is returned. Default is FALSE.

Details

This function calculates the density of the PKB distribution for each data point in y, given the parameters mu and rho.

Value

A vector of density values (or log-density if log = TRUE) for each row in y.

Examples

```
y <- matrix(c(1, 0, 0, 0, 0, 1), ncol = 3, byrow = TRUE)
mu <- c(1, 0, 0)
rho <- 0.5
dpkb(y, mu, rho)
```

`dspcauchy` *Density Function for Spherical Cauchy Distribution*

Description

Calculates the density of the spherical Cauchy distribution for given data points.

Usage

```
dspcauchy(y, mu, rho, log = FALSE)
```

Arguments

`y` A matrix or data frame where each row represents a data point on the unit hypersphere.

`mu` A vector or matrix representing the mean direction parameter(s). If a vector, it must be normalized (unit length) and is applied to all data points. If a matrix, it must have the same number of rows as `y`, and each row must be normalized.

`rho` A scalar or a vector representing the concentration parameter. If a vector, its length must match the number of rows in `y`. Each `rho[i]` is used to evaluate the density for `y[i, ]`. Must be between 0 (inclusive) and 1 (exclusive).

`log` Logical; if TRUE, the log-density is returned. Default is FALSE.

Details

This function calculates the density of the spherical Cauchy distribution for each data point in y, given the parameters mu and rho.

Value

A vector of density values (or log-density if log = TRUE) for each row in y.

Examples

```
y <- matrix(c(1, 0, 0, 0, 0, 1), ncol = 3, byrow = TRUE)
mu <- c(1, 0, 0)
rho <- 0.5
dspcauchy(y, mu, rho)
```

`FLXMCpkb` *PKB distribution Driver for FlexMix*

Description

This model driver for flexmix implements model-based clustering of PKB distributions.

Usage

```
FLXMCpkb(formula = . ~ .)
```

Arguments

`formula` A formula.

Value

Returns an object of class `FLXMC`.

Examples

```
mix <- rbind(rpkb(30, 0.95, c(1, 0, 0)), rpkb(30, 0.9, c(-1, 0, 0)))
m1 <- flexmix::flexmix(mix ~ 1, k = 2, model = FLXMCpkb())
```

FLXMCspcauchy *Spherical Cauchy Driver for FlexMix*

Description

This model driver for flexmix implements model-based clustering of spherical Cauchy distributions.

Usage

```
FLXMCspcauchy(formula = . ~ .)
```

Arguments

formula	A formula.

Value

Returns an object of class FLXMC.

Examples

```
mix <- rbind(rpkb(30, 0.95, c(1, 0, 0)), rpkb(30, 0.9, c(-1, 0, 0)))
m1 <- flexmix::flexmix(mix ~ 1, k = 2, model = FLXMCspcauchy())
```

FLXMRpkb *PKB distribution Driver for FlexMix Using Neural Networks*

Description

This model driver for flexmix implements model-based clustering of PKB distributions using neural network in the M-step.

Usage

```
FLXMRpkb(
  formula = . ~ .,
  EPOCHS = 100,
  LR = 0.1,
  max_iter = 200,
  adam_iter = 5,
  free_iter = adam_iter,
  line_search_fn = "strong_wolfe"
)
```

Arguments

`formula` A formula.

`EPOCHS` The number of epochs in the M-step estimation (default: 100).

`LR` The learning rate used in the M-step estimation (default: 0.1).

`max_iter` The maximum number of iterations of the LBFGS optimizer (default: 200).

`adam_iter` The number of iteration for which the adam optimizer is used before the algorithm switches to L-BFGS (default: 5).

`free_iter` The number of initial iterations for which the model in M-step is fully reseted (default: adam_iter).

`line_search_fn` The method used for line search in LBFGS (default: "strong_wolfe").

Value

Returns an object of class `FLXMC`.

Examples

```
if(torch::torch_is_installed()){
mix <- rbind(rpkb(30, 0.95, c(1, 0, 0)), rpkb(30, 0.9, c(-1, 0, 0)))
m1 <- flexmix::flexmix(mix ~ 1, k = 2, model = FLXMRpkb())
}
```

`FLXMRspcauchy` *Spherical Cauchy Driver for FlexMix Using Neural Networks*

Description

This model driver for flexmix implements model-based clustering of spherical Cauchy distributions using neural networks in the M-step.

Usage

```
FLXMRspcauchy(
  formula = . ~ .,
  EPOCHS = 100,
  LR = 0.1,
  max_iter = 200,
  adam_iter = 5,
  free_iter = adam_iter,
  line_search_fn = "strong_wolfe"
)
```

Arguments

`formula` A formula.

`EPOCHS` EPOCHS The number of epochs in the M-step estimation (default: 100).

`LR` The learning rate used in the M-steo estimation (default: 0.1).

`max_iter` The maximum number of iterations of the LBFGS optimizer (default: 200).

`adam_iter` The number of iteration for which the adam optimizer is used before the algorithm switches to L-BFGS (default: 5).

`free_iter` The number of initial iterations for which the model in M-step is fully reseted (default: adam_iter).

`line_search_fn` The method used for line search in LBFGS (default: "strong_wolfe").

Value

Returns an object of class `FLXMC`.

Examples

```
if(torch::torch_is_installed()){
mix <- rbind(rpkb(30, 0.95, c(1, 0, 0)), rpkb(30, 0.9, c(-1, 0, 0)))
m1 <- flexmix::flexmix(mix ~ 1, k = 2, model = FLXMRspcauchy())
}
```

`rpkb` *Random Sampling from PKB Distributions using ACG or Projected Saw Distributions*

Description

Generates a random sample from PKB distributions.

Usage

```
rpkb(n, rho, mu, method = "ACG")
```

Arguments

`n` The number of random draws.

`rho` A numeric giving the concentration parameter.

`mu` A numeric vector giving the mean direction parameter.

`method` A character indicating the method to use, `"ACG"` for angular central Gaussian distribution envelopes and `"Saw"` for the use of projected Saw distributions.

Value

A matrix with the generated values.

Examples

```
rpkb(10, 0.95, c(1, 0, 0))
```

rspcauchy *Random Sampling from Spherical Cauchy Distributions*

Description

Generates a random sample from spherical Cauchy distributions.

Usage

```
rspcauchy(n, rho, mu)
```

Arguments

n	The number of random draws.
rho	A numeric value giving the rho parameter.
mu	A numeric vector giving the mu direction parameter.

Value

A matrix with the generated values.

Examples

```
rspcauchy(10, 0.95, c(1, 0, 0))
```

Index