Package 'GOFclustering'

May 7, 2026

Type Package

Title Goodness-of-Fit Testing for Model-Based Clustering

Version 1.0.4

Description Performs goodness-of-fit tests for model-based clustering based on the methodology developed in <doi:10.48550/arXiv.2511.04206>. It implements a test statistic derived from empirical likelihood to verify the distribution of clusters in multivariate Gaussian mixtures and other latent class models.

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 4.4)

Imports Rcpp, parallel, VarSelLCM, mixtools, goftest, partitions, randtoolbox, stats, mvtnorm

LinkingTo Rcpp, RcppArmadillo

ByteCompile true

Collate 'GOFclustering.R' 'Bernstein.R' 'fctspsi.R' 'fctsSampler.R' 'Indicator.R' 'RcppExports.R' 'toolsMixtools.R' 'toolsVarSelLCM.R'

RoxygenNote 7.3.3

NeedsCompilation yes

Author Salima El Kolei [aut], Matthieu Marbac [aut, cre]

Maintainer Matthieu Marbac <matthieu.marbac@univ-ubs.fr>

Repository CRAN

Date/Publication 2026-03-21 09:40:02 UTC

Contents

`FitAndGOFCluster` *Fit a Mixture Model and Perform Goodness-of-Fit Test*

Description

High-level function that estimates a clustering model (parametric or non-parametric) and automatically performs the goodness-of-fit test on the posterior probabilities as described in El Kolei and Marbac (2025).

Usage

```
FitAndGOFCluster(
  ech,
  method,
  K,
  kappa = 1/9,
  rho = 3/4,
  xi = 3/4,
  alpha = 0.05,
  basis = "Bernstein",
  size = as.integer(10 * (nrow(ech)^(xi))),
  p = as.integer(round(3 * (nrow(ech)^(kappa)))),
  B = as.integer(round(2 * ((nrow(ech)^(1 - rho)))))
)
```

Arguments

`ech`	A data frame or matrix where each row is an observation.
`method`	Character. The clustering method to use: `"VarSelLCM"` for parametric models or `"Mixtools"` for non-parametric models.
`K`	Integer. The number of clusters to be tested.
`kappa`	Numeric. Tuning parameter for the number of moment conditions `p`. Default is 1/9.
`rho`	Numeric. Tuning parameter for the number of blocks `B`. Default is 3/4.
`xi`	Numeric. Tuning parameter for the sub-sample size `size`. Default is 3/4.
`alpha`	Numeric. Significance level for the test (default is 0.05).
`basis`	Character or vector of characters. The basis used for the moment conditions (`"Bernstein"`, `"indicatorEqualproba"`, or `"indicatorEqualsize"`). If a vector is provided, the test is performed for each basis.
`size`	Integer. Size of the sub-sample for block-splitting. Default is computed based on `xi`.
`p`	Integer. Number of moment conditions. Default is computed based on `kappa`.
`B`	Integer. Number of blocks for the test. Default is computed based on `rho`.

Details

This function automates the estimation and validation. It first fits the model, then calls `GOFClusteringSpe` to compute the test statistics. The test bypasses the curse of dimensionality by working on the unit simplex.

Value

A list containing:

- `clustering`: The fitted model object (`VSLCMresults` or `npEM`).
- `GOF`: A list containing the test results for the first basis:
  - `stat`: The 5 statistics: `max` (Max block EL), `sum` (Sum block EL), `ks` (Kolmogorov-Smirnov), `cvm` (Cramer-von Mises), and `ad` (Anderson-Darling).
  - `pvalues`: P-values for `sum`, `ks`, `cvm`, and `ad`.
  - `reject`: Logical indicators of rejection for each test at level `alpha`.
  - `cvpsi`: The covariance matrix Σ. Its dimension reflects the number of components kept after PCA filtering.
  - `basis`: The basis function used.
  - `hyperparam`: The values of `p` and `B` used.
- `GOF2`: Results for additional bases if provided.

References

El Kolei, S. and Marbac, M. (2025). Goodness-of-fit testing of the distribution of posterior classification probabilities for validating model-based clustering. <doi:10.48550/arXiv.2511.04206>.

Examples

```
# Simulation of   Gaussian Mixture Data (K=2, d=5)
set.seed(123)
n <- 180
z <- sample(1:2, n, replace = TRUE, prob = c(1/2,1/2))
ech <- matrix(0, nrow = n, ncol = 5)
for(k in 1:2) ech[z==k,] <- matrix(rnorm(sum(z==k) * 5, ((-1)**k)/sqrt(5), 1), ncol=5)

# --- Case 1: Parametric estimation and test ---
res_param <- FitAndGOFCluster(ech = ech, method = "VarSelLCM", K = 2, p=3, B=3)
print(res_param$GOF$pvalues)

# --- Case 2: Non-parametric estimation and test ---
res_nonparam <- FitAndGOFCluster(ech = ech, method = "Mixtools", K = 2)
print(res_nonparam$GOF$pvalues)
```

`GOFClustering` *General Goodness-of-Fit Test for Posterior Probabilities*

Description

Core function to perform the goodness-of-fit test for clustering validation. It takes a matrix of posterior probabilities and a function to generate new posterior probabilities under the null hypothesis (estimated model). This version is agnostic to the clustering package used.

Usage

```
GOFClustering(
  probapost,
  rprobapost,
  size = as.integer(10 * (nrow(probapost)^(3/4))),
  p = as.integer(round(3 * (nrow(probapost)^(1/9)))),
  B = as.integer(round(2 * ((nrow(probapost)^(1/4))))),
  alpha = 0.05,
  basis = "Bernstein"
)
```

Arguments

`probapost`	A matrix (n x K) of estimated posterior classification probabilities, where n is the number of observations and K is the number of clusters.
`rprobapost`	A function that generates a new matrix of posterior probabilities under the estimated model. It must take a single argument `n` (sample size).
`size`	Integer. Size of the sub-sample used for the block-splitting procedure. Default is `10 * n^(3/4)`.
`p`	Integer. Number of moment conditions. Default is `3 * n^(1/9)`.
`B`	Integer. Number of blocks for the testing procedure. Default is `2 * n^(1/4)`.
`alpha`	Numeric. Significance level (default is 0.05).
`basis`	Character or vector of characters. The basis used for the moment conditions (`"Bernstein"`, `"indicatorEqualproba"`, or `"indicatorEqualsize"`).

Details

This function implements the Empirical Likelihood test on the unit simplex. The user must provide `rprobapost`, which reflects the uncertainty of the fitted model. For a parametric mixture, this function typically simulates new data from the fitted parameters and returns the resulting posterior probabilities.

Value

A list with the following components:

stat A list of the 5 test statistics: `max`, `sum`, `ks`, `cvm`, and `ad`.

pvalues A list of p-values for `sum`, `ks`, `cvm`, and `ad`.

reject Logical indicators of rejection at level `alpha`.

cvpsi The estimated covariance matrix Σ after PCA filtering.

basis The basis functions used.

hyperparam The number of moments `p` and blocks `B` used.

References

El Kolei, S. and Marbac, M. (2025). Goodness-of-fit testing of the distribution of posterior classification probabilities for validating model-based clustering. <doi:10.48550/arXiv.2511.04206>.

Examples

```
library(VarSelLCM)

# Simulation of a Gaussian Mixture Model (K=2, d=5)
set.seed(123)
n <- 180
z <- sample(1:2, n, replace = TRUE, prob = c(1/2, 1/2))
ech <- matrix(0, nrow = n, ncol = 5)
for(k in 1:2) {
  ech[z==k,] <- matrix(rnorm(sum(z==k) * 5, ((-1)**k)/sqrt(5), 1), ncol=5)
}

# 1. Fit the model using VarSelLCM (without variable selection)
results <- VarSelCluster(ech, 2, vbleSelec = FALSE)

# 2. Define the rprobapost function using the fitted model parameters
# This function simulates data under H0 and computes posterior probabilities
my_rprobapost <- function(n_new) {
  # Simulate labels
  z_sim <- sample(1:results@model@g, n_new, replace = TRUE, results@param@pi)
  # Simulate observations
  ech_sim <- as.data.frame(matrix(NA, n_new, results@data@dataContinuous@d))
  for (j in 1:nrow(results@param@paramContinuous@mu)) {
    for (k in unique(z_sim)) {
      indices <- which(z_sim == k)
      ech_sim[indices, j] <- rnorm(length(indices),
                                   results@param@paramContinuous@mu[j, k],
                                   results@param@paramContinuous@sd[j, k])
    }
  }
  # Compute log-posterior probabilities
  logprobamat <- matrix(log(results@param@pi), n_new, results@model@g, byrow = TRUE)
  for (j in 1:nrow(results@param@paramContinuous@mu)) {
    for (k in 1:ncol(logprobamat)) {
```

```
      logprobamat[, k] <- logprobamat[, k] +
        dnorm(ech_sim[, j], results@param@paramContinuous@mu[j, k],
              results@param@paramContinuous@sd[j, k], log = TRUE)
    }
  }
  # Softmax normalization to obtain probabilities
  probamat <- exp(logprobamat - apply(logprobamat, 1, max))
  probamat <- probamat / rowSums(probamat)
  return(probamat)
}

# 3. Run the general GOF test
test_res <- GOFClustering(probapost = results@partitions@tik,
                          rprobapost = my_rprobapost,
                          p=3,
                          B=3)

# Print results
print(test_res$pvalues)
```

GOFClusteringSpe *Specific Goodness-of-Fit Test for Clustering Objects*

Description

Performs the goodness-of-fit test for a clustering result provided by `VarSelLCM` or `mixtools`. This function implements the block-splitting Empirical Likelihood test on the posterior probabilities.

Usage

```
GOFClusteringSpe(results, size, p, B, alpha = 0.05, basis = "Bernstein")
```

Arguments

`results`	An object of class `VSLCMresults` (from `VarSelLCM`) or `npEM`/`spEMsymloc` (from `mixtools`).
`size`	Integer. Size of the sub-sample for the block-splitting procedure.
`p`	Integer. Number of moment conditions (equations).
`B`	Integer. Number of blocks for the testing procedure.
`alpha`	Numeric. Significance level (default is 0.05).
`basis`	Character or vector of characters. The basis used for the moment conditions (`"Bernstein"`, `"indicatorEqualproba"`, or `"indicatorEqualsize"`).

Details

This function computes the test statistics by splitting the data into `B` blocks to account for parameter estimation uncertainty. The moment conditions are projected using Principal Component Analysis (PCA) to handle potential multicollinearity, especially when `p` is large.

Value

A list with the following components:

stat A list of the 5 test statistics: `max` (Maximum of block EL statistics), `sum` (Sum of block EL statistics), `ks` (Kolmogorov-Smirnov), `cvm` (Cramer-von Mises), and `ad` (Anderson-Darling).

pvalues A list of p-values associated with the statistics (except for `max` which uses a specific critical value).

reject A list of logical indicators. `TRUE` if the null hypothesis (model adequacy) is rejected at level `alpha`.

cvpsi The estimated covariance matrix Σ of the moment conditions. Its dimension may be smaller than `p` due to the PCA filtering.

basis The name of the basis functions used.

hyperparam A list containing the number of moment conditions (`p`) and the number of blocks (`B`).

References

El Kolei, S. and Marbac, M. (2025). Goodness-of-fit testing of the distribution of posterior classification probabilities for validating model-based clustering. <doi:10.48550/arXiv.2511.04206>.

Examples

```
# Simulation of   Gaussian Mixture Data (K=2, d=5)
set.seed(123)
n <- 180
z <- sample(1:2, n, replace = TRUE, prob = c(1/2,1/2))
ech <- matrix(0, nrow = n, ncol = 5)
for(k in 1:2) ech[z==k,] <- matrix(rnorm(sum(z==k) * 5, ((-1)**k)/sqrt(5), 1), ncol=5)

# --- Case 1: Parametric clustering with VarSelLCM ---
library(VarSelLCM)
res_param <- VarSelCluster(ech, 2, vbleSelec = FALSE)
gof_param <- GOFClusteringSpe(res_param, size=500, p=3, B=3)
print(gof_param$pvalues)
print(dim(gof_param$cvpsi)) # Sigma matrix dimension


# --- Case 2: Non-parametric clustering with mixtools ---
library(mixtools)
res_nonparam <- npMSL(ech, mu0 = 2, verb = FALSE)
gof_nonparam <- GOFClusteringSpe(res_nonparam, size=500, p=3, B=3)
print(gof_nonparam$pvalues)
```

Index