Package: RobustLPA (via r-universe)

July 5, 2026

Title Robust Latent Profile Analysis

Version 0.1.0

Description Provides a comprehensive toolset for estimating Latent Profile Analysis (LPA) models that are robust to multivariate outliers and missing data. By integrating a high-performance 'C++' engine via 'RcppArmadillo' with a Full Information Maximum Likelihood (FIML) approach and Huber weighting, it reliably extracts latent profiles even in complex datasets. It supports multiple geometric variance-covariance models, along with functions for bootstrapped likelihood ratio tests and plotting. For methodological details on the Bootstrapped Likelihood Ratio Test, see Nylund et al. (2007) <doi:10.1080/10705510701575396>. For robust clustering methods, see Garcia-Escudero et al. (2010) <doi:10.1007/s11634-010-0064-5>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 8.0.0

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp, ggplot2, stats

NeedsCompilation yes

Author Valerio Riccardo Aquila [aut, cre] (ORCID: <https://orcid.org/0009-0004-2231-2141>)

Maintainer Valerio Riccardo Aquila <valerio_aquila@hotmail.it>

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

Date/Publication 2026-07-05 12:44:49 UTC

RemoteUrl https://github.com/cran/RobustLPA

RemoteRef HEAD

RemoteSha 7820b7e18570355cb291456c9a5f83f4d5a3f130

Contents

`blrt_robust` *Bootstrapped Likelihood Ratio Test for Robust LPA*

Description

Compares a robust LPA model with G profiles against a null model with G-1 profiles using parametric bootstrapping. Supports FIML simulation conditions.

Usage

```
blrt_robust(data, G, model = 6, n_samples = 50, n_starts = 2)
```

Arguments

`data`	A matrix or data.frame.
`G`	The number of profiles for the alternative hypothesis (compared against G-1).
`model`	An integer (1 to 6) specifying the variance-covariance parameterization.
`n_samples`	Number of bootstrap samples. Default is 50 for speed; 200+ is recommended for publications.
`n_starts`	Number of starts for the EM algorithm execution.

Value

A list containing the observed LRT statistic, the vector of bootstrap replicates, and the empirical p-value.

Examples

```
# Fast demonstration of the robust BLRT
data(iris)
blrt_res <- blrt_robust(iris[1:30, 1:2], G = 2, model = 1, n_samples = 2, n_starts = 1)
# Print the summary of the results
blrt_res
```

`estimate_profiles_robust` *Estimate Robust Latent Profile Models*

Description

This function provides a user-friendly interface to fit multiple robust LPA models simultaneously across different numbers of profiles and model structures, returning a comprehensive fit summary table.

Usage

```
estimate_profiles_robust(
  data,
  n_profiles = 1:3,
  models = c(1, 2, 3, 4, 5, 6),
  n_starts = 5
)
```

Arguments

`data`	A matrix or data.frame.
`n_profiles`	A vector of integers specifying the number of profiles to run (e.g., 1:3).
`models`	A vector of LPA models to run (e.g., c(1, 2, 3, 4, 5, 6)). Default is c(1, 2, 3, 4, 5, 6).
`n_starts`	Number of initializations per model.

Value

A list containing the fit comparison table and the estimated models.

Examples

```
# Quick evaluation of multiple profiles
data(iris)
res <- estimate_profiles_robust(iris[1:30, 1:2], n_profiles = 1:2, models = 1, n_starts = 1)
res$fit_table
```

`plot_robust_lpa` *Plot Robust Latent Profiles*

Description

Automatically generates a professional profile plot using ggplot2 from an estimated robust LPA model.

Usage

```
plot_robust_lpa(
  model,
  title = "Robust Latent Profiles",
  xlab = "Variables",
  ylab = "Value",
  var_labels = NULL,
  legend_title = "Class"
)
```

Arguments

`model`	A model object returned by robust_lpa or estimate_profiles_robust.
`title`	The title of the plot. Default is "Robust Latent Profiles".
`xlab`	The x-axis label. Default is "Variables".
`ylab`	The y-axis label. Default is "Value".
`var_labels`	A character vector to manually rename the variables on the X axis. Default is NULL (auto-detect).
`legend_title`	The title of the legend. Default is "Class".

Value

A ggplot object.

Examples

```
data(iris)
fit <- robust_lpa(data = iris[1:30, 1:2], G = 2, model = 1, n_starts = 1)
plot_robust_lpa(fit)
```

robust_lpa *Fit a Single Robust Latent Profile Analysis Model*

Description

This function estimates a single robust Latent Profile Analysis (LPA) model for a specified number of profiles and model structure. It automatically handles missing data via robust FIML if present.

Usage

```
robust_lpa(data, G, model = 6, max_iter = 100, tol = 1e-06, n_starts = 5)
```

Arguments

data	A matrix or data.frame of observations.
G	The number of latent profiles to extract.
model	An integer (1 to 6) specifying the model parameterization.
max_iter	Maximum number of EM iterations.
tol	Tolerance for convergence.
n_starts	Number of random initializations.

Value

A list containing parameters, fit indices, and assignments.

Examples

```
# Quick example using a small subset of the iris dataset
data(iris)
fit <- robust_lpa(data = iris[1:30, 1:2], G = 2, model = 1, n_starts = 1)
# Print the fit indices
fit$fit
```

robust_m_step *Perform a Robust M-Step for Latent Profile Analysis*

Description

Perform a Robust M-Step for Latent Profile Analysis

Usage

```
robust_m_step(data, z, alpha = 0.05)
```

Arguments

data	A matrix or data.frame of observations.
z	A numeric vector containing the posterior probabilities of belonging to this cluster.
alpha	Significance level for the Chi-squared outlier cutoff (default is 0.05).

Value

A list containing the robust 'mean' vector and the robust 'covariance' matrix.

Examples

```
data(iris)
# Simulate initial probabilities for a single profile
z_init <- runif(30)
z_init <- z_init / sum(z_init)

m_step_res <- robust_m_step(iris[1:30, 1:2], z = z_init)

# Print the robust mean calculated in the M-step
m_step_res$mean
```

---

robust_mean *Calculate the robust mean*

Description

Calculate the robust mean

Usage

```
robust_mean(data, threshold = 10)
```

Arguments

data	A matrix or data.frame
threshold	Maximum distance allowed to not be considered an outlier

Value

A numeric vector representing the robust mean of the variables.

Examples

```
data(iris)
r_mean <- robust_mean(iris[1:30, 1:2])

# Print the calculated robust means
r_mean
```

Index