Package 'TICM'

February 11, 2026

Type Package

Title Testing the Validity of the Independent Component Model Assumption

Version 1.0-0

Date 2026-02-09

Maintainer Klaus Nordhausen <klausnordhausenR@gmail.com>

Imports Rcpp (>= 0.11.0), stats, parallel

Depends fICA, JADE, graphics

LinkingTo Rcpp, RcppArmadillo

Description Description: Provides affine-invariant, distribution-free tests of multivariate independence, applied either directly to observed data or to estimated independent components. In the latter case, the procedures can be used to assess the validity of independent component models. The tests are based on L2-type distances between characteristic functions, with inference carried out using permutation or bootstrap resampling schemes. The methods are described in Hallin et al. (2024) <doi:10.48550/arXiv.2404.07632>.

License GPL (>= 3)

NeedsCompilation yes

Author Klaus Nordhausen [aut, cre] (ORCID: <https://orcid.org/0000-0002-3758-8501>), Marc Hallin [aut] (ORCID: <https://orcid.org/0000-0002-6599-7409>), Simos Meintanis [aut] (ORCID: <https://orcid.org/0000-0003-4777-0486>)

Repository CRAN

Date/Publication 2026-02-11 20:20:08 UTC

Contents

TICM-package *Testing the Validity of the Independent Component Model Assumption*

Description

Description: Provides affine-invariant, distribution-free tests of multivariate independence, applied either directly to observed data or to estimated independent components. In the latter case, the procedures can be used to assess the validity of independent component models. The tests are based on L2-type distances between characteristic functions, with inference carried out using permutation or bootstrap resampling schemes. The methods are described in Hallin et al. (2024) <doi:10.48550/arXiv.2404.07632>.

Details

The **TICM** package implements characteristic-function based, L2-type procedures for (i) testing total multivariate independence in observed data and (ii) assessing the validity of an independent component model by testing mutual independence of estimated independent components.

The main user-facing functions are:

- `ind_test`: tests complete independence among the columns of a data matrix.
- `icm_test`: estimates independent components via ICA (e.g., JADE, FOBI, or FastICA) and then performs a total independence test to the estimated components to assess the validity of the independent component model.
- `plot.ticm`: diagnostic visualization of the estimated independent components returned by `icm_test` via a scatterplot matrix.

Both testing functions allow optional marginal score transformations (`"Identity"`, `"Rank"`, `"VdW"`) and offer resampling-based inference using either permutation or bootstrap strategies to approximate p-values.

Typical workflow

1. Use `ind_test` to test whether the observed variables appear mutually independent.
2. Use `icm_test` to estimate independent components and test whether the resulting components are mutually independent (a diagnostic for the independent component model).
3. Use `plot()` on the `ticm` result for a quick visual check of pairwise dependence.

Author(s)

Klaus Nordhausen [aut, cre] (ORCID: <https://orcid.org/0000-0002-3758-8501>), Marc Hallin [aut] (ORCID: <https://orcid.org/0000-0002-6599-7409>), Simos Meintanis [aut] (ORCID: <https://orcid.org/0000-0003-4777-0486>)

Maintainer: Klaus Nordhausen <klausnordhausenR@gmail.com>

References

Hallin, M., Meintanis, S.G. and Nordhausen, K. (2024), *Consistent distribution free affine invariant tests for the validity of independent component models*. arXiv:2404.07632.

See Also

icm_test, ind_test

---

icm_test *Test the Validity of an Independent Component Model*

Description

Performs a test of the independent component model (ICM) assumption. The function first estimates independent components using an ICA method and then tests whether the estimated components are mutually independent. The test statistic is an L2-type functional of empirical characteristic functions, with optional score transformations and resampling-based p-values.

Usage

```
icm_test(X, n.rep = 200, score = "Identity", weight = "Gauss",
         ica = "JADE", strategy = "bootstrap", ncores = NULL,
         iseed = NULL, eps = 1e-06, maxiter = 100, g = "tanh",
         method = "sym", inR = FALSE, n.init = 2)
```

Arguments

`X`	Matrix or data frame of numeric data; rows correspond to observations and columns to variables.
`n.rep`	Number of bootstrap replications or permutations used to approximate the null distribution. Recommended are to perform several thousands.
`score`	Score transformation applied marginally prior to testing: `"Identity"` (raw estimated components), `"Rank"` (componentwise ranks), or `"VdW"` (van der Waerden/normal scores).
`weight`	Weight function used in the L2-type statistic: `"Gauss"` (Gaussian) or `"Laplace"` (Laplace).
`ica`	ICA method used to estimate the mixing/unmixing matrix: `"JADE"`, `"FOBI"`, or `"fICA"`.
`strategy`	Resampling strategy used to obtain p-values: `"bootstrap"` or `"permutation"`.
`ncores`	Number of CPU cores to use for resampling computations. If `NULL`, computation is done sequentially.
`iseed`	Optional integer seed for reproducibility of resampling.
`eps`	Convergence tolerance used by the ICA routines (where applicable).
`maxiter`	Maximum number of iterations used by the ICA routines (where applicable).
`g`	Nonlinearity used by `"fICA"`; see `fICA`. Only used when `ica = "fICA"`.
`method`	FastICA variant: `"sym"`, `"sym2"`, or `"def"`. Only used when `ica = "fICA"`.
`inR`	Logical; if `TRUE`, compute FastICA in R. Only used when `ica = "fICA"`.
`n.init`	Number of random initializations for symmetric FastICA variants. Only used when `ica = "fICA"`.

Details

The independent component model assumes that the observed random vector X can be written as $X = AS$, where A is an invertible mixing matrix and the latent components S have mutually independent coordinates. This function assesses that assumption in two steps:

1. **ICA estimation.** An unmixing matrix is estimated from `X` using the method selected via `ica`. This yields estimated components $\hat{S}$.
2. **Independence testing.** The null hypothesis is that the coordinates of $\hat{S}$ are mutually independent. The test statistic is an L2-type distance between an empirical characteristic function of the joint distribution and the product of the corresponding marginal characteristic functions. The distance is computed with a weight function chosen by `weight`.

Score transformations. To improve robustness and/or obtain distribution-free behavior, the test can be applied not only to $\hat{S}$ itself (`score = "Identity"`), but also after marginal score transformations. With `score = "Rank"`, each component is replaced by its scaled ranks, which yields a test that is invariant under strictly increasing marginal transformations and distribution-free under the null. With `score = "VdW"`, van der Waerden (normal) scores are used; this option is currently available for Gaussian weights.

P-values via resampling. The null distribution of the statistic is approximated by resampling:

- `strategy = "permutation"` breaks cross-component dependence by independently permuting the rows within each component (leaving marginal distributions unchanged).
- `strategy = "bootstrap"` generates a reference distribution using resampling with replacement under an independence structure for the components.

The p-value is computed as the proportion of resampled statistics at least as large as the observed one.

Value

A list of class `"ticm"` and `"htest"` with components including:

`Tboot`	Numeric vector of bootstrap replicates of the test statistic (present when `strategy = "bootstrap"`).
`Tperm`	Numeric vector of permutation replicates of the test statistic (present when `strategy = "permutation"`).
`ICA`	The ICA result applied to `X` (typically an object of class `"bss"`).
`statistic`	The observed value of the test statistic.
`p.value`	Approximate p-value computed from `Tboot` or `Tperm`.
`parameter`	The number of resampling replicates `n.rep`.
`method`	A character string describing the selected ICA method, score type, weight, and resampling strategy.
`data.name`	Name of the input data `X`.
`alternative`	A character string describing the alternative (lack of mutual independence of the estimated components).

References

Hallin, M., Meintanis, S.G., and Nordhausen, K. (2024), *Consistent distribution free affine invariant tests for the validity of independent component models*. arXiv:2404.07632.

See Also

`ind_test`, `fICA`, `JADE`, `FOBI`

Examples

```
n <- 300
A <- matrix(rnorm(9), 3, 3)
s1 <- rt(n, 6)
s2 <- rexp(n, 1)
s3 <- runif(n)
S <- cbind(s1, s2, s3)
X <- S %*% t(A)

# in practice, the number of resamples should be much larger
res <- icm_test(X)
res
plot(res)
```

`ind_test` *Test for Complete Multivariate Independence*

Description

Performs a test of total independence of the components of a multivariate random vector. The test is an L2-type statistic based on characteristic functions, with optional score transformations and resampling-based p-values.

Usage

```
ind_test(X, n.rep = 200, score = "Identity", weight = "Gauss",
         strategy = "bootstrap", ncores = NULL, iseed = NULL)
```

Arguments

`X`	Matrix or data frame of numeric data; rows correspond to observations and columns to variables.
`n.rep`	Number of bootstrap replications or permutations used to approximate the null distribution.
`score`	Score transformation applied marginally to the data: `"Identity"` (raw data), `"Rank"` (componentwise ranks), or `"VdW"` (van der Waerden/normal scores).
`weight`	Weight function used in the L2-type statistic: `"Gauss"` (Gaussian) or `"Laplace"` (Laplace).

`strategy`	Resampling strategy used to obtain p-values: `"bootstrap"` or `"permutation"`.
`ncores`	Number of CPU cores to use for resampling computations. If `NULL`, computation is done sequentially.
`iseed`	Optional integer seed for reproducibility of resampling.

Details

The null hypothesis of the test is that all components of the random vector represented by `X` are mutually independent. The test statistic is constructed as an L2-type distance between the empirical joint characteristic function of the data and the product of the corresponding empirical marginal characteristic functions.

Score transformations. The test may be applied directly to the data (`score = "Identity"`) or after marginal score transformations. With `score = "Rank"`, each marginal variable is replaced by its scaled ranks, leading to a test that is invariant under strictly increasing marginal transformations and distribution-free under the null hypothesis. With `score = "VdW"`, van der Waerden (normal) scores are used to enhance power under a broad class of alternatives.

Weight functions. The L2-type statistic involves integration with respect to a weight function on the characteristic function domain. Gaussian and Laplace weights are available via `weight`. van der Waerden scores are currently implemented only for Gaussian weights.

P-values via resampling. The null distribution of the test statistic is approximated using resampling:

- `strategy = "permutation"` independently permutes the observations within each variable, thereby destroying cross-component dependence while preserving marginal distributions.
- `strategy = "bootstrap"` generates a reference distribution using resampling with replacement under an independence structure.

The p-value is computed as the proportion of resampled statistics at least as large as the observed one.

Value

An object of class `"htest"` containing:

`statistic`	The observed value of the test statistic.
`p.value`	Approximate p-value based on resampling.
`method`	A character string describing the test, including score, weight, and resampling strategy.
`data.name`	Name of the input data `X`.
`alternative`	A character string describing the alternative hypothesis (lack of mutual independence).
`parameter`	The number of resampling replicates `n.rep`.

References

Hallin, M., Meintanis, S.G. and Nordhausen, K. (2024), *Consistent distribution free affine invariant tests for the validity of independent component models*. arXiv:2404.07632.

See Also

`icm_test`

Examples

```
# in practice, the number of resamples should be much larger
n <- 100
X <- matrix(rnorm(n * 3), ncol = 3)

res <- ind_test(X)
res
```

`plot.ticm` *Plot Estimated Independent Components from a `ticm` Object*

Description

Produces a scatterplot matrix of the estimated independent components stored in a `ticm` object, typically returned by `icm_test`.

Usage

```
## S3 method for class 'ticm'
plot(x, ...)
```

Arguments

`x`	An object of class `ticm`, typically resulting from a call to `icm_test`.
`...`	Additional graphical parameters passed to `pairs` to customize the scatterplot matrix (e.g., `pch`, `cex`, `gap`, `labels`).

Details

The function extracts the matrix of estimated independent components from `x` (as obtained from the ICA step within `icm_test`) and displays a scatterplot matrix using `pairs`. This visualization is useful for a quick diagnostic of remaining dependence: under a well-fitting independent component model, pairwise plots of the estimated components should show little visible structure.

The argument `...` is forwarded to `pairs`, allowing customization of points, labels, and panel functions (e.g., adding smoothers or correlations via `panel =` or `upper.panel =`).

Value

No return value, called for side effects.

See Also

`icm_test`, `pairs`

Examples

```
n <- 200
X <- cbind(runif(n), rexp(n), rnorm(n))

# in practice, the number of resamples should be much larger
res <- icm_test(X)
plot(res)
```

Index