Package ‘MDMICA’

February 8, 2018

Title Independent Component Analysis via Mutual Dependence Measures

Version 1.0.0

Date 2018-01-30

Description Implementation of independent component analysis methods based on mutual dependence measures in Jin, Z., and Matteson, D. S. (2017) <https://arxiv.org/abs/1709.02532> and Pfister, N., et al. (2018) <doi:10.1111/rssb.12235>.

Depends R (>= 3.4.0)

Imports energy (>= 1.7-2),
MDMeasure (>= 1.0.0),
dHSIC (>= 2.0),
rBayesianOptimization (>= 1.1.0)

Suggests testthat (>= 2.0.0)

License GPL (>= 2)

LazyData true

RoxygenNote 6.0.1

R topics documented:

`MDMICA-package` *Independent Component Analysis via Mutual Dependence Measures*

Description

MDMICA: A package for independent component analysis via mutual dependence measures

Details

The MDMICA package provides independent component analysis methods based on mutual dependence measures.

Applying mutual dependence measures

The mutual dependence measures include:

- distance-based energy statistics
  - asymmetric measure $\mathcal{R}_n$ based on distance covariance $\mathcal{V}_n$
  - symmectric measure $\mathcal{S}_n$ based on distance covariance $\mathcal{V}_n$
  - simplified complete measure $\mathcal{Q}_n^\star$ based on incomplete V-statistics
- kernel-based maximum mean discrepancies
  - d-variable Hilbert–Schmidt independence criterion dHSIC$_n$ based on Hilbert–Schmidt independence criterion HSIC$_n$

Initializing local optimization methods

The initialization methods include:

- Latin hypercube sampling
- Bayesian optimization

Author(s)

Ze Jin <zj58@cornell.edu>, David S. Matteson <matteson@cornell.edu>

---

mdm_ica — *Independent Component Analysis via Mutual Dependence Measures*

Description

mdm_ica performs independent component analysis by minimizing mutual dependence measures of all univariate components in X.

Usage

```
mdm_ica(X, num_lhs = NULL, mdm_type = "comp", num_bo = NULL,
  kernel = "exp", opt_algo = "par")
```

Arguments

X	A matrix or data frame, where rows represent samples, and columns represent components.
num_lhs	The number of points generated by Latin hypercube sampling. If omitted, an adaptive number is used.
mdm_type	The type of mutual dependence measures, including • asym asymmetric measure $\mathcal{R}_n$ based on distance covariance $\mathcal{V}_n$; • sym: symmectric measure $\mathcal{S}_n$ based on distance covariance $\mathcal{V}_n$; • comp: simplified complete measure $\mathcal{Q}_n^\star$ based on incomplete V-statistics; • dhsic: d-variable Hilbert–Schmidt independence criterion dHSIC$_n$ based on Hilbert–Schmidt independence criterion HSIC$_n$.
num_bo	The number of points evaluated by Bayesian optimization.

kernel — The kernel of the underlying Gaussian process in Bayesian optimization, including

- exp: squared exponential kernel;
- mat: Matern 5/2 kernel.

opt_algo — The algorithm of optimization, including

- def: deflation algorithm, where the components are extracted one at a time;
- par: parallel algorithm, where the components are extracted simultaneously.

Value

mdm_ica returns a list including the following components:

theta	The rotation angles of the estimated unmixing matrix.
W	The estimated unmixing matrix.
obj	The objective value of the estimated independence components.
S	The estimated independence components.

References

Jin, Z., and Matteson, D. S. (2017). Generalizing Distance Covariance to Measure and Test Multivariate Mutual Dependence. arXiv preprint arXiv:1709.02532. https://arxiv.org/abs/1709.02532.

Pfister, N., et al. (2018). Kernel-based tests for joint independence. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 80(1), 5-31. http://dx.doi.org/10.1111/rssb.12235.

Examples

```
# X is a 10 x 3 matrix with 10 samples and 3 components
X <- matrix(rnorm(10 * 3), 10, 3)

# deflation algorithm
mdm_ica(X, mdm_type = "asym", opt_algo = "def")
# parallel algorithm
mdm_ica(X, mdm_type = "asym", opt_algo = "par")

## Not run:
# bayesian optimization with exponential kernel
mdm_ica(X, mdm_type = "sym", num_bo = 1, kernel = "exp", opt_algo = "par")
# bayesian optimization with matern kernel
mdm_ica(X, mdm_type = "comp", num_bo = 1, kernel = "mat", opt_algo = "par")

## End(Not run)
```

Index