

Package: groupICA (via r-universe)

September 6, 2024

Title Independent Component Analysis for Grouped Data

Version 0.1.1

Author Niklas Pfister and Sebastian Weichwald

Maintainer Niklas Pfister <pfister@stat.math.ethz.ch>

Description Contains an implementation of an independent component analysis (ICA) for grouped data. The main function `groupICA()` performs a blind source separation, by maximizing an independence across sources and allows to adjust for varying confounding for user-specified groups. Additionally, the package contains the function `uwedge()` which can be used to approximately jointly diagonalize a list of matrices. For more details see the project website <<https://sweichwald.de/groupICA/>>.

URL <https://github.com/sweichwald/groupICA-R>

BugReports <https://github.com/sweichwald/groupICA-R/issues>

Depends R (>= 3.2.3)

License AGPL-3

Encoding UTF-8

LazyData true

Imports stats, MASS

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-06-19 08:55:29 UTC

Contents

groupICA	2
uwedge	4
Index	7

groupICA

*groupICA***Description**

Estimates the unmixing and confounded sources of the groupICA model $X=A(S+H)$.

Usage

```
groupICA(X, group_index = NA, partition_index = NA, n_components = NA,
         n_components_uwedge = NA, rank_components = FALSE,
         pairing = "complement", groupsize = 1, partitionsize = NA,
         max_iter = 1000, tol = 1e-12, silent = TRUE)
```

Arguments

<code>X</code>	data matrix. Each column corresponds to one predictor variable.
<code>group_index</code>	vector coding to which group each sample belongs, with $\text{length}(\text{group_index})=\text{nrow}(X)$. If no group index is provided a rigid grid with <code>groupsize</code> samples per group is used (which defaults to all samples if <code>groupsize</code> was not set).
<code>partition_index</code>	vector coding to which partition each sample belongs, with $\text{length}(\text{partition_index})=\text{nrow}(X)$. If no partition index is provided a rigid grid with <code>partitionsize</code> samples per partition is used.
<code>n_components</code>	number of components to extract. If NA is passed, the same number of components as the input has dimensions is used.
<code>n_components_uwedge</code>	number of components to extract during uwedge approximate joint diagonalization of the matrices. If NA is passed, the same number of components as the input has dimensions is used.
<code>rank_components</code>	boolean, optional. When TRUE, the components will be ordered in decreasing stability.
<code>pairing</code>	either 'complement' or 'allpairs'. If 'allpairs' the difference matrices are computed for all pairs of partition covariance matrices, while if 'complement' a one-vs-complement scheme is used.
<code>groupsize</code>	int, optional. Approximate number of samples in each group when using a rigid grid as groups. If NA is passed, all samples will be in one group unless <code>group_index</code> is passed during fitting in which case the provided group index is used (the latter is the advised and preferred way).
<code>partitionsize</code>	int, optional. Approximate number of samples in each partition when using a rigid grid as partition. If NA is passed, a (hopefully sane) default is used, again, unless <code>partition_index</code> is passed during fitting in which case the provided partition index is used.

max_iter	int, optional. Maximum number of iterations for the uwedge approximate joint diagonalisation during fitting.
tol	float, optional. Tolerance for terminating the uwedge approximate joint diagonalisation during fitting.
silent	boolean whether to suppress status outputs.

Details

For further details see the references.

Value

object of class 'GroupICA' consisting of the following elements

V	the unmixing matrix.
covered	boolean indicating whether the approximate joint diagonalisation converged due to tol.
n_iter	number of iterations of the approximate joint diagonalisation.
meanoffdiag	mean absolute value of the off-diagonal values of the to be jointly diagonalised matrices, i.e., a proxy of the approximate joint diagonalisation objective function.

Author(s)

Niklas Pfister and Sebastian Weichwald

References

Pfister, N., S. Weichwald, P. Bühlmann and B. Schölkopf (2017). GroupICA: Independent Component Analysis for grouped data. ArXiv e-prints (arXiv:1806.01094).

Project website (<https://sweichwald.de/groupICA/>)

See Also

The function `uwedge` allows to perform to perform an approximate joint matrix diagonalization.

Examples

```
## Example
set.seed(1)

# Generate data from a block-wise variance model
d <- 2
m <- 10
n <- 5000
group_index <- rep(c(1,2), each=n)
partition_index <- rep(rep(1:m, each=n/m), 2)
S <- matrix(NA, 2*n, d)
H <- matrix(NA, 2*n, d)
```

```

for(i in unique(group_index)){
  varH <- abs(rnorm(d))/4
  H[group_index==i, ] <- matrix(rnorm(d*n)*rep(varH, each=n), n, d)
  for(j in unique(partition_index[group_index==i])){
    varS <- abs(rnorm(d))
    index <- partition_index==j & group_index==i
    S[index,] <- matrix(rnorm(d*n/m)*rep(varS, each=n/m),
                      n/m, d)
  }
}
A <- matrix(rnorm(d^2), d, d)
A <- A%%t(A)
X <- t(A%%t(S+H))

# Apply groupICA
res <- groupICA(X, group_index, partition_index, rank_components=TRUE)

# Compare results
par(mfrow=c(2,2))
plot((S+H)[,1], type="l", main="true source 1", ylab="S+H")
plot(res$Shat[,1], type="l", main="estimated source 1", ylab="Shat")
plot((S+H)[,2], type="l", main="true source 2", ylab="S+H")
plot(res$Shat[,2], type="l", main="estimated source 2", ylab="Shat")
cor(res$Shat, S+H)

```

uwedge

uwedge

Description

Performs an approximate joint matrix diagonalization on a list of matrices. More precisely, for a list of matrices R_x the algorithm finds a matrix V such that for all i $V R_x[i] t(V)$ is approximately diagonal.

Usage

```
uwedge(Rx, init = NA, rm_x0 = TRUE, return_diag = FALSE, tol = 1e-10,
       max_iter = 1000, n_components = NA, silent = TRUE)
```

Arguments

<code>Rx</code>	list of matrices to be diagonalized.
<code>init</code>	matrix used in first step of initialization. If NA a default based on PCA is used
<code>rm_x0</code>	boolean whether to also diagonalize first matrix in R_x or only use it for scaling.
<code>return_diag</code>	boolean. Specifies whether to return the list of diagonalized matrices.
<code>tol</code>	float, optional. Tolerance for terminating the iteration.
<code>max_iter</code>	int, optional. Maximum number of iterations.
<code>n_components</code>	number of components to extract. If NA is passed, all components are used.
<code>silent</code>	boolean whether to suppress status outputs.

Details

For further details see the references.

Value

object of class 'uwedge' consisting of the following elements

V	joint diagonalizing matrix.
Rxdiag	list of diagonalized matrices.
converged	boolean specifying whether the algorithm converged for the given tol.
iterations	number of iterations of the approximate joint diagonalisation.
meanoffdiag	mean absolute value of the off-diagonal values of the to be jointly diagonalised matrices, i.e., a proxy of the approximate joint diagonalisation objective function.

Author(s)

Niklas Pfister and Sebastian Weichwald

References

Pfister, N., S. Weichwald, P. Bühlmann and B. Schölkopf (2017). GroupICA: Independent Component Analysis for grouped data. ArXiv e-prints (arXiv:1806.01094).

Tichavsky, P. and Yeredor, A. (2009). Fast Approximate Joint Diagonalization Incorporating Weight Matrices. IEEE Transactions on Signal Processing.

See Also

The function [groupICA](#) uses uwedge.

Examples

```
## Example
set.seed(1)

# Generate data 20 matrix that can be jointly diagonalized
d <- 10
A <- matrix(rnorm(d*d), d, d)
A <- A%*%t(A)
Rx <- lapply(1:20, function(x) A %*% diag(rnorm(d)) %*% t(A))

# Perform approximate joint diagonalization
ptm <- proc.time()
res <- uwedge(Rx,
              rm_x0=FALSE,
              return_diag=TRUE,
              max_iter=1000)
print(proc.time()-ptm)
```

```
# Average value of offdiagonal elements:  
print(res$meanoffdiag)
```

Index

groupICA, [2](#), [5](#)

uwedge, [3](#), [4](#)