Package ‘CICA’

May 7, 2026

Type Package

Title Clusterwise Independent Component Analysis

Version 1.1.1

Date 2024-07-11

Depends ica, RNifti, Rfast, R (>= 4.3)

Imports mclust, plotly, multiway, methods, magrittr, neurobase, oro.nifti, servr, htmltools

Author Jeffrey Durieux [aut, cre], Tom Wilderjans [aut], Juan Claramunt Gonzalez [ctb]

Maintainer Jeffrey Durieux `<durieux.jeffrey@gmail.com>`

Description

Clustering multi-subject resting state functional Magnetic Resonance Imaging data. This methods enables the clustering of subjects based on multi-subject resting state functional Magnetic Resonance Imaging data. Objects are clustered based on similarities and differences in cluster-specific estimated components obtained by Independent Component Analysis.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

URL https://www.sciencedirect.com/science/article/pii/S0165027022002448

NeedsCompilation no

Repository CRAN

Date/Publication 2024-07-14 15:20:13 UTC

Contents

`CICA` *CICA: Clusterwise Independent Component Analysis*

Description

Main function to perform Clusterwise Independent Component Analysis

Usage

```
CICA(
  DataList,
  nComp,
  nClus,
  method = "fastICA",
  RanStarts,
  RatStarts = NULL,
  pseudo = NULL,
  pseudoFac,
  userDef = NULL,
  userGrid = NULL,
  scalevalue = 1000,
  center = TRUE,
  maxiter = 100,
  verbose = TRUE,
  ctol = 1e-06,
  checks = TRUE
)
```

Arguments

`DataList`	a list of matrices
`nComp`	number or vector of ICA components per cluster
`nClus`	number or vector of clusters
`method`	Component method, default is `fastICA`. `EVD` for a fast eigen value based estimation
`RanStarts`	number of random starts
`RatStarts`	Generate rational starts. Either 'all' or a specific linkage method name (e.g., 'complete'). Use NULL to indicate that Rational starts should not be used.
`pseudo`	percentage value for perturbating rational starts to obtain pseudo rational starts
`pseudoFac`	factor to multiply the number of rational starts (7 in total) to obtain pseudorational starts
`userDef`	a user-defined starting seed stored in a data.frame, if NULL no userDef starting partition is used
`userGrid`	user supplied data.frame for multiple model CICA. First column are the requested components. Second column are the requested clusters
`scalevalue`	desired sum of squares of the block scaling procedure
`center`	mean center matrices
`maxiter`	maximum number of iterations for each start
`verbose`	print loss information to console
`ctol`	tolerance value for convergence criterion
`checks`	boolean parameter that indicates whether the input checks should be run (TRUE) or not (FALSE).

Value

`CICA` returns an object of `class` "CICA". It contains the estimated clustering, cluster specific component matrices and subject specific time course matrices

`P`	partitioning vector of size `length(DataList)`
`Sr`	list of size `nClus`, containing cluster specific independent components
`Ais`	list of size `length(DataList)`, containing subject specific time courses
`Loss`	loss function value of the best start
`FinalLossDiff`	value of the loss difference between the last two iterations of the algorithm.
`IndLoss`	a vector with containing the individual loss function values
`LossStarts`	loss function values of all starts
`Iterations`	Number of iterations
`starts`	dataframe with the used starting partitions

Author(s)

Jeffrey Durieux

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

multiple_output = CICA(DataList = CICA_data$X, nComp = 2:6, nClus = 1:5,
method = 'fastICA',userGrid = NULL, RanStarts = 30, RatStarts = NULL,
pseudo = c(0.1, 0.2),pseudoFac = 2, userDef = NULL, scalevalue = 1000,
center = TRUE,maxiter = 100, verbose = TRUE, ctol = .000001)

summary(multiple_output$Q_5_R_4)

plot(multiple_output$Q_5_R_4)

## End(Not run)
```

`computeRVmat` *Compute modified RV matrix*

Description

This function computes a NxN modified RV matrix

Usage

```
computeRVmat(DataList = DataList, dist = TRUE, verbose = TRUE)
```

Arguments

`DataList`	a list with matrices
`dist`	boolean if TRUE distance object is returned
`verbose`	boolean if TRUE progressbar is printed to the console

Value

`RVsS`	a square similarity matrix of `class matrix` or distance object of `class dist` containing the pairwise modified RV values

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)
#Compute single subject ICAs (nClus equals length(ExampleData))
output <- CICA(DataList = CICA_data$X, nStarts = 1,
               nComp = 5, nClus = 9, verbose = FALSE)
```

```
RV <- computeRVmat(DataList = output$Sr, dist = TRUE,
                   verbose = FALSE)

# apply hierarchical clustering on RV output
hcl <- hclust(RV)
plot(hcl)

# low dimensional visualisation using Classical Multidimensional Scaling
mds <- cmdscale(RV)
plot(mds)

## End(Not run)
```

`embed_papaya` *Embed images with Papaya*

Description

Writes temporary images out from nifti objects or passes character filenames of images to papaya JS viewer

Usage

```
embed_papaya(images, outdir = NULL)
```

Arguments

`images`	character filenames or `nifti` objects to be viewed
`outdir`	output directory for index and all to go

Value

Output html

`FindRationalStarts` *Plot method for rstarts object*

Description

Two step clustering for finding rational start partitions

Usage

```
FindRationalStarts(
  DataList,
  RatStarts = "all",
  nComp,
  nClus,
  scalevalue = NULL,
  center = TRUE,
  verbose = TRUE,
  pseudo = NULL,
  pseudoFac = NULL
)

## S3 method for class 'rstarts'
plot(x, type = 1, mdsdim = 2, nClus = NULL, ...)
```

Arguments

`DataList`	a list of matrices
`RatStarts`	type of rational start. 'all' computes all types of hclust methods
`nComp`	number of ICA components to extract
`nClus`	Number of clusters for rectangles in dendrogram, default NULL is based on number of clusters present in the object
`scalevalue`	scale each matrix to have an equal sum of squares
`center`	mean center matrices
`verbose`	print output to console
`pseudo`	percentage value for perturbating rational starts to obtain pseudo rational starts
`pseudoFac`	how many pseudo starts per rational start
`x`	an object of `class` rstarts
`type`	type of plot, 1 for a dendrogram, 2 for a multidimensional scaling configuration
`mdsdim`	2 for two dimensional mds configuration, 3 for a three dimensional configuration
`...`	optional arguments passed to `hclust` function

Value

dataframe with (pseudo-) rational and dist object based on the pairwise modified RV values

References

Durieux, J., & Wilderjans, T. F. (2019). Partitioning subjects based on high-dimensional fMRI data: comparison of several clustering methods and studying the influence of ICA data reduction in big data. Behaviormetrika, 46(2), 271-311.

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)
rats <- FindRationalStarts(DataList = CICA_data$X, nComp = 5, nClus = 4,verbose = TRUE, pseudo = .2)
plot(rats, type = 1, method = 'ward.D2')
plot(rats, type = 2, method = 'ward.D2')
plot(rats, type = 2, method = 'ward.D2', mdsdim = 3)

## End(Not run)


## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)
Out_starts <- FindRationalStarts(DataList = CICA_data$X,nComp = 5,nClus = 4,scalevalue = 1000)
plot(Out_starts)
plot(Out_starts, type = 2)
plot(Out_starts, type = 2,mdsdim = 3, method = 'ward.D2')

## End(Not run)
```

`GenRanStarts` *Generate random starts*

Description

Generate random starts

Usage

```
GenRanStarts(
  RanStarts,
  nClus,
  nBlocks,
  ARIlim = 0.2,
  itmax = 1000,
  verbose = FALSE
)
```

Arguments

`RanStarts`	number of randomstarts to generate
`nClus`	number of clusters
`nBlocks`	number of objects
`ARIlim`	maximal value of adjusted Rand Index

`itmax`	maximum number of iterations used to find suitable random starts
`verbose`	boolean that indicates whether the output should be printed on the console

Value

a list where the first element is a matrix with random starts, second element all pairwise ARIs #'

`GenRatStarts` *Title*

Description

Title

Usage

```
GenRatStarts(
  DataList,
  RatStarts,
  nComp,
  nClus,
  scalevalue,
  center,
  verbose,
  pseudo,
  pseudoFac
)
```

Arguments

`DataList`	DataList
`RatStarts`	Type of rational start
`nComp`	number of components
`nClus`	number of clusters
`scalevalue`	value for blockscaling procedure
`center`	center
`verbose`	verbose
`pseudo`	percentage used for perturbation rational starts (between 0)
`pseudoFac`	multiplication factor for pseudo rational starts

Value

out

`get_papaya_version` *Get Papaya Version*

Description

Reads the papaya.js file installed and determines version and build

Usage

```
get_papaya_version()
```

Value

List of build and version, both characters

`loadNIfTIs` *Load Nifti files from directory*

Description

Load Nifti files from directory

Usage

```
loadNIfTIs(dir, toMatrix = TRUE)
```

Arguments

`dir`	Input directory containing nifti files
`toMatrix`	logical if TRUE nifti's are converted to matrices

Value

list object containing Voxel by Time course matrices

Examples

```
## Not run:
nifs <- loadNIfTIs('<FolderPath>', toMatrix = T)
outnif <- CICA(DataList = nifs, RanStarts = 2, nComp = 10, nClus = 2)

## End(Not run)
```

`matcher` *Match components between cluster specific spatial maps*

Description

Match components between cluster specific spatial maps

Usage

```
matcher(x, reference, RV = FALSE, ...)
```

Arguments

`x`	object of class CICA
`reference`	integer cluster index that serves as the reference. If nifti path is supplied, clusters will be matched to this template
`RV`	compute modified-RV between cluster components
`...`	other arguments

Value

out

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

multiple_output = CICA(DataList = CICA_data$X, nComp = 2:6, nClus = 1:5,
userGrid = NULL, RanStarts = 30, RatStarts = NULL, pseudo = c(0.1, 0.2),
pseudoFac = 2, userDef = NULL, scalevalue = 1000, center = TRUE,
maxiter = 100, verbose = TRUE, ctol = .000001)

matcher(multiple_output$Q_5_R_4, reference = 1, RV = TRUE)

## End(Not run)
```

`matcher.CICA` *Match components between cluster specific spatial maps*

Description

Match components between cluster specific spatial maps

Usage

```
## S3 method for class 'CICA'
matcher(x, reference = 1, RV = FALSE, ...)
```

Arguments

`x`	object of class CICA
`reference`	integer cluster index that serves as the reference. If nifti path is supplied, clusters will be matched to this template
`RV`	compute modified-RV between cluster components
`...`	other arguments

Value

out

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

multiple_output = CICA(DataList = CICA_data$X, nComp = 2:6, nClus = 1:5,
userGrid = NULL, RanStarts = 30, RatStarts = NULL, pseudo = c(0.1, 0.2),
pseudoFac = 2, userDef = NULL, scalevalue = 1000, center = TRUE,
maxiter = 100, verbose = TRUE, ctol = .000001)

matcher(multiple_output$Q_5_R_4, reference = 1, RV = TRUE)

## End(Not run)
```

`mpinv` *Moore Penrose inverse*

Description

Moore Penrose inverse

Usage

```
mpinv(X)
```

Arguments

`X`	input matrix

Value

mp Moore Penrose inverse of matrix X

`papaya` *View images with Papaya*

Description

Writes temporary images out from nifti objects or passes character filenames of images to papaya JS viewer

Usage

```
papaya(images, outdir = NULL, ...)
```

Arguments

`images`	character filenames or `nifti` objects to be viewed
`outdir`	output directory for index and all to go
`...`	Options to be passed to `pass_papaya`

Value

Output directory where index.html, js, and copied nii.gz files

Examples

```
## Not run:
library(neurobase)
x = nifti(img = array(rnorm(100^3), dim= rep(100, 3)), dim=rep(100, 3), datatype=16)
thresh = datatyper(x > 1)
index.file = papaya(list(x, thresh))

## End(Not run)
```

`papaya_div` *Papaya Div element output*

Description

Get the necessary div output for embedding a papaya image

Usage

```
papaya_div()
```

Value

Character string

Examples

```
papaya_div()
```

`pass_papaya` *View images with Papaya*

Description

Writes temporary images out from nifti objects or passes character filenames of images to papaya JS viewer

Usage

```
pass_papaya(
  L = NULL,
  outdir = NULL,
  daemon = FALSE,
  close_on_exit = TRUE,
  sleeper = 3,
  version = "0.8",
  build = "982"
)
```

Arguments

`L`	list of arguments passed to papaya using params
`outdir`	output directory for index and all to go
`daemon`	Argument passed to `server_config`
`close_on_exit`	Should the server close once the function finishes?
`sleeper`	Time in seconds to sleep if `close_on_exit` = `TRUE`. This allows the server to start up.
`version`	Version of papaya.js and papaya.css to use
`build`	Build of papaya.js and papaya.css to use

`plot.CICA` *Plot method for CICA*

Description

Plot method for CICA. This function shows the cluster specific independent components in an interactive viewer using the papayar package

Usage

```
## S3 method for class 'CICA'
plot(x, brain = "auto", cluster = 1, ...)
```

Arguments

`x`	Object of class CICA
`brain`	auto
`cluster`	Components of cluster to plot. Only used when non fMRI related data is used
`...`	other arguments

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

multiple_output = CICA(DataList = CICA_data$X, nComp = 2:6, nClus = 1:5,
userGrid = NULL, RanStarts = 30, RatStarts = NULL, pseudo = c(0.1, 0.2),
pseudoFac = 2, userDef = NULL, scalevalue = 1000, center = TRUE,
maxiter = 100, verbose = TRUE, ctol = .000001)

plot(multiple_output$Q_5_R_4, cluster = 2)

## End(Not run)
```

`plot.ModSel` *Plot method for sequential model selection*

Description

Plot method for the sequential model selection option for CICA

Usage

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

Arguments

`x`	Object of `class` ModSel
`...`	other arguments

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

multiple_output = CICA(DataList = CICA_data$X, nComp = 2:6, nClus = 1:5,
userGrid = NULL, RanStarts = 30, RatStarts = NULL, pseudo = c(0.1, 0.2),
pseudoFac = 2, userDef = NULL, scalevalue = 1000, center = TRUE,
maxiter = 100, verbose = TRUE, ctol = .000001)

ModSelOutput <- SequentialScree(multiple_output)

plot(ModSelOutput)

## End(Not run)
```

`SequentialScree` *Sequential Model Selection for Multiple CICA model*

Description

Sequential Model Selection for Multiple CICA model

Usage

```
SequentialScree(x)
```

Arguments

`x`	an object of class MultipleCICA

Value

a list object

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

multiple_output = CICA(DataList = CICA_data$X, nComp = 2:6, nClus = 1:5,
userGrid = NULL, RanStarts = 30, RatStarts = NULL, pseudo = c(0.1, 0.2),
pseudoFac = 2, userDef = NULL, scalevalue = 1000, center = TRUE,
maxiter = 100, verbose = TRUE, ctol = .000001)

ModSelOutput <- SequentialScree(multiple_output)

plot(ModSelOutput)

## End(Not run)
```

`Sim_CICA` *Simulate CICA data*

Description

Simulate CICA data

Usage

```
Sim_CICA(
  Nr,
  Q,
  R,
  voxels,
  timepoints,
  E,
  overlap = NULL,
  externalscore = FALSE
)
```

Arguments

`Nr`	number of subjects per cluster
`Q`	number of components
`R`	number of clusters
`voxels`	number of voxels
`timepoints`	number of time points
`E`	proportion of independent gaussian noise
`overlap`	amount of overlap between S across clusters. Smaller value means more overlap
`externalscore`	add simulated external score (default is FALSE)

Value

a list with simulated CICA data

Examples

```
## Not run:

#Use set.seed(1) to obtain the dataset used in the article "Clusterwise
#Independent Component Analysis (CICA): an R package for clustering subjects
#based on ICA patterns underlying three-way (brain) data"

Xe <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

## End(Not run)
```

`Sr_to_nifti` *Convert Cluster specific independent components to NIFTI format*

Description

Convert Cluster specific independent components to NIFTI format

Usage

```
Sr_to_nifti(x, write = FALSE, ...)
```

Arguments

`x`	an object of `class` CICA
`write`	if TRUE, NIfTI files are written to current working directory
`...`	other arguments passed to RNifti::writeNifti

Value

a list with niftiImage files

Examples

```
## Not run:
nifs <- loadNIfTIs('<FolderPath>', toMatrix = T)
outnif <- CICA(DataList = nifs, RanStarts = 2, nComp = 10, nClus = 2)
test <- Sr_to_nifti(outnif,write = T, datatype = 'int16', version = 2)

## End(Not run)
```

`summary.CICA` *Summary method for class CICA*

Description

Summarize a CICA analysis

Usage

```
## S3 method for class 'CICA'
summary(object, ...)
```

Arguments

`object`	Object of the type produced by `CICA`
`...`	Additional arguments

Value

`summary.CICA` returns an overview of the estimated clustering of a `CICA` analysis

`PM`	Partitioning matrix
`tab`	tabulation of the clustering
`Loss`	Loss function value of the solution

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

multiple_output = CICA(DataList = CICA_data$X, nComp = 2:6, nClus = 1:5,
userGrid = NULL, RanStarts = 30, RatStarts = NULL, pseudo = c(0.1, 0.2),
pseudoFac = 2, userDef = NULL, scalevalue = 1000, center = TRUE,
maxiter = 100, verbose = TRUE, ctol = .000001)
```

```
summary(multiple_output$Q_5_R_4)

## End(Not run)
```

`summary.MultipleCICA` *Summary method for class MultipleCICA*

Description

Summarize a CICA analysis

Usage

```
## S3 method for class 'MultipleCICA'
summary(object, ...)
```

Arguments

`object`	Object of the type produced by `CICA`
`...`	Additional arguments

Value

`summary.MultipleCICA` returns an overview of the estimated clustering of a `CICA` analysis

`PM`	Partitioning matrix
`tab`	tabulation of the clustering
`Loss`	Loss function value of the solution

Examples

```
## Not run:
CICA_data <- Sim_CICA(Nr = 15, Q = 5, R = 4, voxels = 100, timepoints = 10,
E = 0.4, overlap = .25, externalscore = TRUE)

multiple_output = CICA(DataList = CICA_data$X, nComp = 2:6, nClus = 1:5,
userGrid = NULL, RanStarts = 30, RatStarts = NULL, pseudo = c(0.1, 0.2),
pseudoFac = 2, userDef = NULL, scalevalue = 1000, center = TRUE,
maxiter = 100, verbose = TRUE, ctol = .000001)

summary(multiple_output$Q_5_R_4)

## End(Not run)
```

`update_papaya_build` *Update Papaya build version from GitHub*

Description

Updates the papaya version in the papayar package to the most current on GitHub

Usage

```
update_papaya_build(
  type = c("standard", "minimal", "nodicom", "nojquery", "standard-with-atlas-local",
     "standard-with-atlas"),
   verbose = TRUE
)
```

Arguments

`type`	Type of release. Standard is default
`verbose`	Should download progress be shown?

Value

Result of `get_papaya_version` after downloading

Index