Package ‘omicade4’
February 1, 2017

Type Package
Title Multiple co-inertia analysis of omics datasets
Version 1.15.0
Date 2015-06-10
Author Chen Meng, Aedin Culhane, Amin M. Gholami.
Maintainer Chen Meng <mengchen18@gmail.de>
Imports made4
Depends R (>= 3.0.0), ade4
Suggests BiocStyle
Description Multiple co-inertia analysis of omics datasets
License GPL-2
LazyLoad yes
biocViews Software, Clustering, Classification, MultipleComparison
NeedsCompilation no

R topics documented:

omicade4-package ....................................................... 2
mcia ................................................................. 3
NCI60_4arrays .......................................................... 5
plotVar ............................................................... 6
plotVar.cia ............................................................. 7
plotVar.mcia .......................................................... 8
selectVar ............................................................ 9
selectVar.cia .......................................................... 10
selectVar.mcia ......................................................... 11
topVar ............................................................... 12
topVar.cia ............................................................. 13
topVar.mcia .......................................................... 13
The main function in the package performing multiple co-inertia analysis on omics datasets.

Multiple co-inertia analysis (MCIA) is a multivariate analysis method that could be used to analyze multiple tables measuring the same set of individuals, this package provides a one-stop function for MCIA and functions for subsequent analysis especially for multiple omics datasets.

Chen Meng, Aedin Culhane, Amin M. Gholami
Maintainer: Chen Meng <mengchen18@gmail.com>

Meng C, Kuster B, Culhane AC and Gholami AM. A multivariate approach to the integration of multi-omics datasets. (Manuscript under preparation)


See Also
ade4 and package made4

Examples
```
data(NCI60_4arrays)
mcoin <- mcia(NCI60_4arrays)
```
Description

The main function in omicade4. Performing multiple co-inertia analysis on a list of data.frames or matrix.

Usage

```r
cmpia(df.list, cia.nf = 2, cia.scan = FALSE, nsc = T, svd = TRUE)
```

## S3 method for class 'mcia'

```r
plot(x, axes = 1:2,
     sample.lab = TRUE, sample.legend = TRUE, sample.color = 1,
     phenovec = NULL, df.color = 1,
     df.pch = NA, gene.nlab = 0, ...)
```

Arguments

- `df.list`: A list of `data.frame`s, matrix or ExpressionSet is going to be analyzed, the column number must be the same and mapped across all `data.frame`/matrix.
- `cia.nf`: An integer indicating the number of kept axes.
- `cia.scan`: A logical indicating whether the co-inertia analysis eigenvalue (scree) plot should be shown so that the number of axes `(cia.nf)` can be selected interactively. Default value is FALSE.
- `nsc`: A logical indicating whether multiple co-inertia analysis should be performed using multiple non-symmetric correspondence analyses `dudi.nsc`. The default = TRUE is highly recommended. If FALSE, COA `dudi.coa` will be performed on the first `data.frame`, and row weighted COA `dudi.rwcoa` will be performed on the rest ones using the row weights from the first one.
- `svd`: A logical indicates which function should be used to perform singular value decomposition.
- `sample.lab`: A logical indicating if the samples should be labelled, the default is TRUE.
- `sample.color`: Defining colours of samples for plotting sample space, the length of this argument should be either one (uniform color) or the same with the column number of `data.frame` in `df.list`.
- `sample.legend`: A logical indicating if the legend for sample space should be drawn.
- `df.color`: Defining the colours for plotting variables (genes) from different `data.frame`. The length of this argument should be either one (all datasets use the same colour) or the same number of datasets (each dataset has a specified colour, the repetitive use of colour code is allowed.)
- `df.pch`: Defining the `pch` for plotting variable (gene) space. The default is NA, the function will distinguish datasets by default. Otherwise, the length of this argument should be either one (all datasets use the same `pch`) or the same number of datasets (each dataset has a specified `pch`).
- `phenovec`: A factor for plotting sample space, phenovec could be used to distinguish individuals in the `data.frame`.
- `x`: An object of class `mcia`
axes A vector of integer in length 2 to indicate the axes are going to be plotted. The default are first two axes.
gene.nlab An integer indicating how many top weighted genes on each axis should be labelled

Other arguments

Details

The column number of data.frame in the df.list must be the same, and the same column from different data.frame should be matchable. For example, Microarray profiling for the same set of cell lines, patients and etc.

mcia calls dudi.nsc, ktab and mcoa in ade4 packages.

Plotting and visualizing mcia results

Two functions could be used to visualize the result of mcia: The first is plot.mcia, which results in four plots. Top left represents the sample space. Individuals from the same column of different data.frames are linked by edges. Different platforms are distinguished by the shape of points. Top right shows the variable space, datasets are marked by different colours. Bottom left represents the eigenvalue scree plot. The pseudo-eigenvalue space of all data.frames are visualized in the bottom right panel. The second function is plotVar.mcia, which could be used to plot the variable space for different datasets as well as finding and visualizing the variables (genes) across datasets.

Other methods

selectVar.mcia: selecting variables (genes) according to the their coordinates.

Value

call the function called
mcoa The results returned by mcoa
coa The results returned by separate analysis (applying dudi.nsc or dudi.coa on each data.frame separately)

Author(s)

Chen Meng

See Also

See Also as mcoa, plotVar, plotVar

Examples

data(NCI60_4arrays)
mcoin <- mcia(NCI60_4arrays)
plot(mcoin, sample.lab=FALSE, df.col=4:7)

colcode <- sapply(strsplit(colnames(NCI60_4arrays$agilent), split="\."),
function(x) x[1])
plot(mcoin, sample.lab=FALSE, sample.color=as.factor(colcode))
Microarray gene expression profiles of the NCI 60 cell lines from 4 different platforms

Description
The 60 human tumour cell lines are derived from patients with leukaemia, melanoma, lung, colon, central nervous system, ovarian, renal, breast and prostate cancers. The cell line panel is widely used in anti-cancer drug screen. In this dataset, a subset of microarray gene expression of the NCI 60 cell lines from four different platforms are combined in a list, which could be used as input to mcia directly.

Usage
data(NCI60_4arrays)

Format
The format is: List of 4 data.frames

- \$agilent: data.frame containing 300 rows and 60 columns. 300 gene expression log ratio measurements of the NCI60 cell lines, by Agilent platform.
- \$hgu133: data.frame containing 298 rows and 60 columns. 298 gene expression log ratio measurements of the NCI60 cell lines, by H-GU133 platform.
- \$hgu133p2: data.frame containing 268 rows and 60 columns. 268 gene expression log ratio measurements of the NCI60 cell lines, by H-GU133 plus 2.0 platform.
- \$hgu95: data.frame containing 288 rows and 60 columns. 288 gene expression log ratio measurements of the NCI60 cell lines, by H-GU95 platform.

Source

References

Examples

data(NCI60_4arrays)
summary(NCI60_4arrays)
mcoin <- mcia(NCI60_4arrays)
plotVar

Plot variable (gene) spaces of result from MCIA or CIA

Description
The user level function for plotting variable space of mcia or cia, which could be used to visualize selected variables (genes) across datasets. It calls plotVar.cia or plotVar.mcia.

Usage
```r
plotVar(x, var = NA, axes = 1:2, var.col = "red", var.lab = FALSE, bg.var.col = "gray", nlab = 0, sepID.data=NULL, sepID.sep=".", ...)
```

Arguments
- **x**: An object of class cia or mcia
- **var**: A character vector defining the variables (genes) are going to be labelled and coloured. The default NA means no variables (genes) selected.
- **axes**: An integer vector in length 2 indicating which axes are going to be plotted. Default are the first two axes.
- **var.col**: The colour of selected variables (genes), the length of this argument should be either 1 (uniform colour) or the length of var (each var has a specified colour).
- **var.lab**: A logical indicating if the variables (genes) selected should be labelled, the default is FALSE.
- **bg.var.col**: Colour code for unselected variables (genes) in all datasets.
- **nlab**: An integer indicating how many top weighted genes on each axis should be labelled.
- **sepID.data**: This argument enables a more generalized mapping of identifiers in different datasets. For example, if there is a PTM (post-transcriptional modification) dataset in one of the data.frames, the corresponding protein could be detected with setting this argument. For more details, see "details" section.
- **sepID.sep**: Used to help determine the separator of variables (genes) in the sepID.data. For more details, see "details" section.
- **...**: Other arguments

Details
For the sepID.data, a typical example is the post-transcriptional modification (PTM) data. The name of variables (genes) have a general form like "proteinName_modificationSite". The sepID.data specifies the IDs from dataset that should be separated, sepID.sep specifies the separator of protein name and modification site. This is used to determine the same proteins/genes across different datasets.

Value
If var is not NA, a data frame is returned, with rows for variables (genes) of interest and columns of logical values indicating which dataset contains which variables (genes).
plotVar.cia

Author(s)

Chen Meng

See Also

See Also as `plotVar.cia, plotVar.mcia`

Examples

```r
data(NCI60_4arrays)
mcoin <- mcia(NCI60_4arrays)
plotVar(mcoin, var=c("S100B", "S100A1"), var.lab=TRUE)

# an example for the usage of sepID.data and sepID.sep
nci60_mod <- NCI60_4arrays
rownames(nci60_mod$hgu95) <- paste(rownames(nci60_mod$hgu95), "s1", sep="_")
mcoin_mod <- mcia(nci60_mod)
# without specifying
plotVar(mcoin_mod, var=c("S100B", "S100A1"), var.lab=TRUE)
# specifying the sepID.data and sepID.sep
plotVar(mcoin_mod, var=c("S100B", "S100A1"), var.lab=TRUE, sepID.data=4, sepID.sep="_")
```

plotVar.cia  
Plot variable space of result from `cia`

Description

Plot variable space of `mcia` and visualize selected variables across datasets.

Usage

```r
## S3 method for class 'cia'
plotVar(x, var = NA, axes = 1:2, 
var.col = "red", var.lab = FALSE, bg.var.col = "gray", 
nlab = 0, sepID.data = NULL, sepID.sep = "_", ...
```

Arguments

- `x`: An object of class `cia`
- `var`: see `plotVar`
- `axes`: see `plotVar`
- `var.col`: see `plotVar`
- `var.lab`: see `plotVar`
- `bg.var.col`: see `plotVar`
- `nlab`: see `plotVar`
- `sepID.data`: see `plotVar`
- `sepID.sep`: see `plotVar`
- ...: Other arguments
plotVar.mcia

**Value**

If `var` is not NA, a data frame is return, with rows for variables of interest and columns of logical value indicating which data.frames contains which variables.

**Author(s)**

Chen Meng

**See Also**

See Also as `plotVar.mcia`

---

**plotVar.mcia**

*Plot variable space of result from mcia*

**Description**

Plot variable space of `mcia` and visualize selected variables across datasets, the function is called by `plotVar`.

**Usage**

```r
## S3 method for class 'mcia'
plotVar(x, var = NA, axes = 1:2,
       var.col = "red", var.lab = FALSE, bg.var.col = "gray",
       nlab = 0, sepID.data = NULL, sepID.sep = ",",
       df = NA, layout = NA, ...)
```

**Arguments**

- `x` An object of class `mcia`, the result returned by `mcia`.
- `var` see `plotVar`
- `axes` see `plotVar`
- `var.col` see `plotVar`
- `var.lab` see `plotVar`
- `bg.var.col` see `plotVar`
- `nlab` see `plotVar`
- `sepID.data` see `plotVar`
- `sepID.sep` see `plotVar`
- `df` Integers indicating which dataset should be plotted, the default NA means all datasets are plotted.
- `layout` The layout of multiple plots.
- `...` Other arguments

**Value**

If `var` is not NA, a data frame is return, with rows for variables of interest and columns of logical values indicating which data.frames contains which variables.
selectVar

Author(s)
Chen Meng

See Also
See Also as `plotVar.cia, plotVar`

Examples

```r
data(NCI60_4arrays)
mcoin <- mcia(NCI60_4arrays)
plot.mcia(mcoin, sample.lab=FALSE, df.col=4:7)
plotVar(mcoin, var=NA, bg.var.col=1:4, var.lab=TRUE)
plotVar(mcoin, var=c("SP0PL", "CAPN2", "SNX8"),
        df=1:4, var.lab=TRUE, var.col=c("red", "green", "blue"))
```

**selectVar**  
*Selecting variables (genes) from result of MCIA or CIA according to co-ordinates*

Description
The user level function calls `selectVar.mcia` or `selectVar.cia`. Function `cia` or `mcia` projects variables (genes) from different datasets to a 2 dimensional space. This function supplies a method selecting variables (genes) according to the coordinates of variables.

Usage

```
selectVar(x, axis1 = 1, axis2 = 2, ...)
```

Arguments
- `x` : An object of class `cia` or `mcia`, the result returned by `cia` or `mcia` respectively.
- `axis1` : Integer, the column number for the x-axis. The default is 1.
- `axis2` : Integer, the column number for the y-axis. The default is 2.
- `...` : Other arguments

Value
Returns a data.frame describing which variables (genes) are presented on which data.frames within the limited region(s).

Author(s)
Chen Meng

See Also
See Also as `selectVar.mcia, selectVar.cia`
Examples

```r
data(NCI60_4arrays)
mcoin <- mcia(NCI60_4arrays)
selectVar(mcoin, a1.lim=c(2, Inf), a2.lim=c(-Inf, Inf))

# an example for the usage of sepID.data and sepID.sep
nci60_mod <- NCI60_4arrays
rownames(nci60_mod$hgu95) <- paste(rownames(nci60_mod$hgu95), "s1", sep="_")
mcoin_mod <- mcia(nci60_mod)
# without specifying
selectVar(mcoin_mod, a1.lim=c(2, Inf), a2.lim=c(-Inf, Inf))
# specifying the sepID.data and sepID.seq
selectVar(mcoin_mod, a1.lim=c(2, Inf), a2.lim=c(-Inf, Inf), sepID.data=4, sepID.seq="_")
```

selectVar.cia

Selecting variables from result of CIA

Description

To select variables in CIA variable space, the function is called by selectVar.

Usage

```r
## S3 method for class 'Var.cia'
selectVar(x, axis1 = 1, axis2 = 2,
  df1.a1.lim = c(-Inf, Inf), df1.a2.lim = c(-Inf, Inf),
  df2.a1.lim = df1.a1.lim, df2.a2.lim = df1.a2.lim,
  sepID.data = NULL, sepID.sep = "_", ...)
```

Arguments

- **x** The result returned by `cia`
- **axis1** Integer, the column number for the x-axis. The default is 1.
- **axis2** Integer, the column number for the y-axis. The default is 2.
- **df1.a1.lim** A vector containing 2 numbers indicating the range of X axis of selected on the 1st data.frame. The first value limiting the lower boundary, the second value limiting the upper boundary.
- **df1.a2.lim** The range of Y axis of selected on the 1st datasets.
- **df2.a1.lim** The range of X axis of selected on the 2nd dataset.
- **df2.a2.lim** The range of Y axis of selected on the 2nd dataset.
- **sepID.data** See `plotVar.mcia`
- **sepID.sep** See `plotVar.mcia`
- **...** Other arguments

Details

cia projecting variables from different datasets to a two dimensional space. This function supplies a method selecting variables according to the co-ordinates of variables.
selectVar.mcia

Value

Returns a data.frame describing which variables are presented on which data.frame within the limited region(s).

Author(s)

Chen Meng

See Also

See Also as selectVar.mcia

selectVar.mcia  Selecting variables from result of MCIA

Description

The selection of variables based on co-ordinates of MCIA variable space. The function is called by selectVar

Usage

## S3 method for class 'mcia'
selectVar(x, axis1 = 1, axis2 = 2,
a1.lim = c(-Inf, Inf), a2.lim = c(-Inf, Inf),
sepID.data = NULL, sepID.sep = "_", ...)

Arguments

x
An object of class mcia, the result returned by mcia.

axis1
Integer, the column number for the x-axis. The default is 1.

axis2
Integer, the column number for the y-axis. The default is 2.

a1.lim
The limited range of x-axis of selected. It could be either a vector (containing 2 numbers, the first value limiting the lower boundary, the second value limiting the upper boundary) or a list of vectors, each of which contains two number. If it is a list, the length of the list should be the same with number of data.frames in mcia.

a2.lim
The limited range of y-axis.

sepID.data
See plotVar.mcia

sepID.sep
See plotVar.mcia

...
Other arguments

Details

mcia projecting variables (genes) from different datasets to a lower dimensional space. This function supplies a method selecting variables according to the co-ordinates of variables.
Value

Returns a data.frame describing which variables are presented on which data.frames within the limited region(s).

Author(s)

Chen Meng

See Also

See Also as `selectVar.cia, selectVar`

Examples

data(NCI60_4arrays)
mcoin <- mcia(NCI60_4arrays)
selectVar(mcoin, a1.lim=c(1, Inf))

topVar

Selecting top weighted variables (genes) from result of MCIA or CIA

Description

The user level function calls `topVar.mcia` or `topVar.cia`. This function provides a method selecting top weighted variables (genes) on an axis (either positive side or negative side or both).

Usage

topVar(x, axis = 1, end = "both", topN = 5)

Arguments

x an object of class mcia or cia
axis an integer to specify which axis to check
end which end of the axis to check, could be positive, negative or both. Any unambiguous substring can be given.
topN An integer. The number of top weighted variable to return.

Value

Returns a data.frame contains selected variables.

Author(s)

Chen Meng

Examples

data(NCI60_4arrays)
mcoin <- mcia(NCI60_4arrays)
topVar(mcoin, axis = 1, end = "both", topN = 3)
Selecting top weighted variables (genes) from result of CIA

Description

This function provides a method selecting top weighted variables (genes) on an axis (either positive side or negative side or both) from an object of class cia (see made4 package).

Usage

```r
## S3 method for class 'cia'
topVar(x, axis = 1, end = "both", topN = 5)
```

Arguments

- `x` See `plotVar.mcia`
- `axis` See `plotVar.mcia`
- `end` See `plotVar.mcia`
- `topN` See `plotVar.mcia`

Value

See `plotVar.mcia`

Author(s)

Chen Meng

Selecting top weighted variables (genes) from result of MCIA

Description

This function provides a method selecting top weighted variables (genes) on an axis (either positive side or negative side or both) from an object of class mcia.

Usage

```r
## S3 method for class 'mcia'
topVar(x, axis = 1, end = "both", topN = 5)
```

Arguments

- `x` See `plotVar.mcia`
- `axis` See `plotVar.mcia`
- `end` See `plotVar.mcia`
- `topN` See `plotVar.mcia`
topVar.mcia

Value
See plotVar.mcia

Author(s)
Chen Meng
Index

*Topic Microarray
   NCI60_4arrays, 5

*Topic NCI-60
   NCI60_4arrays, 5

*Topic datasets
   NCI60_4arrays, 5

*Topic mcia
   mcia, 3

*Topic multivariate
   omicade4-package, 2

ade4, 2

cia, 6, 7, 9, 10

dudi.coa, 3, 4
dudi.nsc, 3, 4
dudi.rwcoa, 3

ktab, 4

mcia, 3, 6–9, 11
mcoa, 4

NCI60_4arrays, 5

omicade4 (omicade4-package), 2
omicade4-package, 2

plot.mcia(mcia), 3
plotVar, 6, 7–9
plotVar.cia, 7, 7, 9
plotVar.mcia, 7, 8, 8, 10, 11, 13, 14

selectVar, 9, 12
selectVar.cia, 9, 10, 12
selectVar.mcia, 9, 11, 11

topVar, 12
topVar.cia, 13
topVar.mcia, 13