Package ‘InPosition’

January 31, 2019

Type Package
Title Inference Tests for ExPosition
Version 0.12.7.1
Date 2013-12-09
Author Derek Beaton, Joseph Dunlop, Herve Abdi
Maintainer Derek Beaton <exposition.software@gmail.com>
Description Non-parametric resampling-based inference tests for ExPosition.
License GPL-2
Depends prettyGraphs (>= 2.1.4), ExPosition (>= 2.0.0)
BugReports http://code.google.com/p/exposition-family/issues/list
NeedsCompilation no
Repository CRAN
Date/Publication 2019-01-31 17:27:41 UTC

R topics documented:

  InPosition-package ........................................... 2
  boot.compute.fj ........................................... 3
  boot.ratio.test ........................................... 4
  boot.samples .............................................. 6
  caChiTest .................................................. 7
  contingency.data.break ................................... 8
  continueResampling ....................................... 9
  epCA.inference.battery .................................... 9
  epGPCA.inference.battery .................................. 11
  epMCA.inference.battery ................................... 12
  epPCA.inference.battery ................................... 14
  inGraphs .................................................... 16
  print.epCA.inference.battery .............................. 17
  print.epGPCA.inference.battery ......................... 18
  print.epMCA.inference.battery ........................... 18
InPosition-package

print.epPCA.inference.battery ........................................ 19
print.inpoBoot ................................................................. 19
print.inpoBootTests ...................................................... 20
print.inpoComponents .................................................... 20
print.inpoOmni ............................................................... 21
print.inpoOutput ............................................................ 21
rebuildMCAtable ............................................................ 22

Index 23

InPosition-package

InPosition-package

InPosition: Inference Tests for Exploratory Analysis with the Singular Value DecomPosition (ExPosition).

Description

InPosition provides multiple forms of inference tests for the ExPosition package.

Details

Package: InPosition
Type: Package
Version: 0.12.7
Date: 2013-12-09
Depends: R (>=2.15.0), prettyGraphs (>= 2.1.4), ExPosition (>= 2.0.0)
License: GPL-2
URL: http://www.utdallas.edu/~derekbeaton/software/ExPosition

Author(s)

Questions, comments, compliments, and complaints go to Derek Beaton <exposition.software@gmail.com>. Also see the bug-tracking and live update website for ExPosition: http://code.google.com/p/exposition-family/

Primary authors and contributors are: Derek Beaton, Joseph Dunlop, and Hervé Abdi

References

Permutation:
Bootstrap:

See Also

- `epPCA.inference.battery`, `epGPCA.inference.battery`, `epCA.inference.battery`, `epMCA.inference.battery`

There are no inference tests for MDS at this time. We recommend PCA for inference instead of MDS (some MDS inference tests require the rectangluar table, not the distances, so it is easier to just use PCA).

See also `inGraphs` for graphing and `caChiTest` for an alternate to resampling methods for Correspondence Analysis.

Examples

```r
# For more examples, see each individual function (as noted above).
```

```r
boot.compute.fj

*Compute bootstrap resampled fj as supplemental elements.*
```

Description

This function computes a bootstrap resampled set of data and projects fj as supplemental elements.

Usage

```r
boot.compute.fj(DATA, res, DESIGN = NULL, constrained = FALSE)
```

Arguments

- **DATA**
  The original data matrix to be bootstrapped. Rows will be bootstrapped and are assumed to be observations.
- **res**
  of class `expoOutput`. Results from one of the `ExPosition` methods (e.g., `epPCA`, `epMCA`),
- **DESIGN**
  A design matrix (in disjunctive coding). Only used if `constrained` is TRUE.
- **constrained**
  a boolean. If TRUE, bootstrap resampling will occur within groups as designated by the DESIGN matrix.

Value

- **fjj**
  a set of factor scores of the measures (columns, fj) for the bootstrapped data.
Author(s)

Derek Beaton

References


See Also

See the functions supplementaryCols and link{bootNsamples}

Examples

```r
# the following code generates 100 bootstrap resampled
# projections of the measures from the Iris data set.
data(epNiris)
data < ep.iris$data
design <- ep.iris$design
iris.pca <- epPCA(data, scale="SS1", DESIGN=design, make_design_nominal=FALSE)
boot.fjs.unconstrained <- array(0, dim=c(dim(iris.pca$ExPosition.Data$fj), 100))
boot.fjs.constrained <- array(0, dim=c(dim(iris.pca$ExPosition.Data$fj), 100))
for(i in 1:100){
  # unconstrained means we resample any of the 150 flowers
  boot.fjs.unconstrained[,i] <- boot.compute.fj(ep.iris$data, iris.pca)
  # constrained resamples within each of the 3 groups
  boot.fjs.constrained[,i] <- boot.compute.fj(data, iris.pca, design, TRUE)
}
```

boot.ratio.test performs bootstrap ratio test.

Description

Performs bootstrap ratio test which is analogous to a t- or z-score.

Usage

```r
boot.ratio.test(boot.cube, critical.value = 2)
```
boot.ratio.test

Arguments

boot.cube an array. This is the bootstrap resampled data. dim 1 (rows) are the items to be tested (e.g., fj, see boot.compute.fj). dim 2 (columns) are the components from the supplemental projection. dim 3 (depth) are each bootstrap sample.
critical.value numeric. This is the value that would be used as a cutoff in a t- or z-test. Default is 2 (i.e., 1.96 rounded up). The higher the number, the more difficult to reject the null.

Value

A list with the following items:
return(list(sig.boot.ratios=significant.boot.ratios,boot.ratios=boot.ratios,critical.value=critical.value))

sig.boot.ratios This is a matrix with the same number of rows and columns as boot.cube. If TRUE, the bootstrap ratio was larger than critical.value. If FALSE, it was smaller.
boot.ratios This is a matrix with bootstrap ratio values that has the same number of rows and columns as boot.cube.
critical.value the critical value input is also returned.

Author(s)

Derek Beaton and Hervé Abdi

References


The bootstrap ratio is related to other tests of values with respect to the bootstrap distribution, such as the Interval-t. See: Chernick, M. R. (2008). Bootstrap methods: A guide for practitioners and researchers (Vol. 619). Wiley-Interscience.

See Also

boot.compute.fj

Examples

```r
# the following code generates 100 bootstrap resampled projections of the measures from the Iris data set.
data(ep.iris)
data <- ep.iris$data
```
design <- ep.iris$design
iris.pca <- epGPCA(data, scale="SS1", DESIGN=design, make_design_nominal=FALSE)

boot.fjs.unconstrained <- array(0, dim=c(dim(iris.pca$exPosition.Data$fj), 100))
boot.fjs.constrained <- array(0, dim=c(dim(iris.pca$exPosition.Data$fj), 100))

for(i in 1:100){
  # unconstrained means we resample any of the 150 flowers
  boot.fjs.unconstrained[,,i] <- boot.compute.fj(ep.iris$data, iris.pca)
  # constrained resamples within each of the 3 groups
  boot.fjs.constrained[,,i] <- boot.compute.fj(data, iris.pca, design, TRUE)
}

# now compute the bootstrap ratios:
ratios.unconstrained <- boot.ratio.test(boot.fjs.unconstrained)
ratios.constrained <- boot.ratio.test(boot.fjs.constrained)

---

**boot.samples**

*Compute indices for bootstrap resampling.*

**Description**

This function computes a set of indices for bootstrap resampling. It can be unconstrained or bootstrap within a group design.

**Usage**

```r
boot.samples(DATA, DESIGN = NULL, constrained = FALSE)
```

**Arguments**

- **DATA**
  - The original data matrix to be bootstrapped. Rows will be bootstrapped and are assumed to be observations.

- **DESIGN**
  - A design matrix (in disjunctive coding). Only used if constrained is TRUE.

- **constrained**
  - A boolean. If TRUE, bootstrap resampling will occur within groups as designated by the DESIGN matrix.

**Value**

- A set of indices to be used as the bootstrap resampled indices.

**Author(s)**

Derek Beaton

**See Also**

- `boot.compute.fj` and `boot.ratio.test`
caChiTest

Examples

data(ep.iris)
unconstrained.indices <- boot.samples(ep.iris$data)
#ep.iris$data[unconstrained.indices]
constrained.indices <- boot.samples(ep.iris$data, DESIGN=ep.iris$design, constrained=TRUE)
#ep.iris$data[constrained.indices]

cachitest

cachitest: correspondence analysis tests without resampling.

Description

cachTest performs 3 sets of chi-square tests along the lines of Lebart’s v-tests. These tests are
designed to be conservative estimates of chi-square tests on contingency data. The tests treat this
data in a standard chi-square framework, but are helpful to understand correspondence analysis data
when permutation and bootstrap become unfeasible.

Usage

cachTest(DATA, res, critical.value = 2)

Arguments

DATA Data as would be entered for Correspondence Analysis (see link(epCA))
res Results from correspondence analysis (e.g., output from link(epCA)).
critical.value numeric. A value, analogous to a z- or t-score to be used to determine signifi-
cance (via bootstrap ratio).

Value

a list with the following values:

j.sig.vals boolean matrix. Identifies which column items are significant (based on critical.value).
j.signed.vals chi-square values associated to column items, multiplied by the sign of their component scores ($f_j$).
j.p.vals p values associated to column items in a chi-square test.
i.sig.vals boolean matrix. Identifies which row items are significant (based on critical.value).
i.signed.vals chi-square values associated to row items, multiplied by the sign of their component scores ($f_i$).
i.p.vals p values associated to row items in a chi-square test.
omni.val chi-square value associated to the table.
omni.p p value associated to a chi-square tests of the table.
Bootstrap or permutation resampling for contingency tables

Description

Bootstrap or permutation resampling for contingency tables. More specifically, for correspondence analysis (epCA).

Usage

contingency.data.break(DATA, boot = FALSE)

Arguments

- **DATA**: A contingency table to resample.
- **boot**: a boolean. If TRUE, use bootstrap (resample with replacement) resampling. If FALSE, use permutation (resample with no replacement).

Value

A resampled contingency table.

Author(s)

Joseph Dunlop and Derek Beaton

See Also

epCA, epCA.inference.battery

Examples

data(authors)
boot.authors <- contingency.data.break(authors$ca$data,boot=TRUE)
perm.authors <- contingency.data.break(authors$ca$data)
**continueResampling**

A stopping mechanism if resampling will take too long.

**Description**

This function asks the user if they want to continue with resampling if the total time for resampling takes more than 1 minute. It also provides an estimate of how long resampling takes. This function is required for InPosition and TinPosition and we do not recommend others use it.

**Usage**

`continueResampling(cycle.time)`

**Arguments**

- `cycle.time`: Is the subtraction of two calls to `proc.time`.

**Note**

If computation time is expected to take more than 1 minute and `interactive()` is TRUE, this asks the user if they would like to continue. If 'Y', looping continues. If 'N', it stops.

If computation time is expected to take more than 1 minute and `interactive()` is FALSE, the function will proceed as is and perform inference tests.

A progress bar is provided so the user can see how long the tests will take.

See inference battery functions for details.

**Author(s)**

Derek Beaton

---

**epCA.inference.battery**

`epCA.inference.battery`: Inference tests for Correspondence Analysis (CA) via InPosition.

**Description**

Correspondence Analysis (CA) and a battery of inference tests via InPosition. The battery includes permutation and bootstrap tests.

**Usage**

`epCA.inference.battery(DATA, DESIGN = NULL, make_design_nominal = TRUE, masses = NULL, weights = NULL, hellinger = FALSE, symmetric = TRUE, graphs = TRUE, k = 0, test.iters = 100, critical.value = 2)`
**Arguments**

- **DATA**: original data to perform a CA on.
- **DESIGN**: a design matrix to indicate if rows belong to groups.
  - **make_design_nominal**: a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.
- **masses**: a diagonal matrix or column-vector of masses for the row items.
- **weights**: a diagonal matrix or column-vector of weights for the column it
- **hellinger**: a boolean. If FALSE (default), Chi-square distance will be used. If TRUE, Hellinger distance will be used.
- **symmetric**: a boolean. If TRUE (default) symmetric factor scores for rows and columns are computed. If FALSE, the simplex (column-based) will be returned.
- **graphs**: a boolean. If TRUE (default), graphs and plots are provided (via `epGraphs`).
- **k**: number of components to return.
- **test.iters**: number of iterations
- **critical.value**: numeric. A value, analogous to a z- or t-score to be used to determine significance (via bootstrap ratio).

**Details**

`epCA.inference.battery` performs correspondence analysis and inference tests on a data matrix.

If the expected time to compute the results (based on `test.iters`) exceeds 1 minute, you will be asked (via command line) if you want to continue.

**Value**

Returns two lists (`$Fixed.Data` and `$Inference.Data`). For `$Fixed.Data`, see `epCA, coreCA` for details on the descriptive (fixed-effects) results.

-$Inference.Data$ returns:
- **components**: Permutation tests of components. p-values (`$p.vals`) and distributions of eigenvalues (`$eigs.perm`) for each component
- **fj.boots**: Bootstrap tests of measures (columns). See `boot.ratio.test` output details.
- **omni**: Permutation tests of components. p-values (`$p.val`) and distributions of total inertia (`$inertia.perm`)

**Author(s)**

Derek Beaton, Joseph Dunlop, and Hervé Abdi.

**See Also**

`epCA, epMCA, epMCA.inference.battery, caChiTest`
epGPCA.inference.battery

**Examples**

```r
## warning: this example takes a while to compute. This is why it is reduced.
data(authors)ca.authors.res <- epCA.inference.battery(authors$ca$data/100)
```

epGPCA.inference.battery

epGPCA.inference.battery: Inference tests for Generalized Principal Component Analysis (PCA) via InPosition.

**Description**

Generalized Principal Component Analysis (PCA) and a battery of inference tests via InPosition. The battery includes permutation and bootstrap tests.

**Usage**

```r
epGPCA.inference.battery(DATA, scale = TRUE, center = TRUE, DESIGN = NULL,
make_design_nominal = TRUE, masses = NULL, weights = NULL,
graphs = TRUE, k = 0,
test.iters = 100, constrained = FALSE, critical.value = 2)
```

**Arguments**

- **DATA** original data to perform a PCA on.
- **scale** a boolean, vector, or string. See `expo.scale` for details.
- **center** a boolean, vector, or string. See `expo.scale` for details.
- **DESIGN** a design matrix to indicate if rows belong to groups.
- **make_design_nominal** a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.
- **masses** a diagonal matrix or column-vector of masses for the row items.
- **weights** a diagonal matrix or column-vector of weights for the column items.
- **graphs** a boolean. If TRUE (default), graphs and plots are provided (via `epGraphs`)
- **k** number of components to return.
- **test.iters** number of iterations
- **constrained** a boolean. If a DESIGN matrix is used, this will constrain bootstrap resampling to be within groups.
- **critical.value** numeric. A value, analogous to a z- or t-score to be used to determine significance (via bootstrap ratio).
epMCA.inference.battery

Details

epGPCA.inference.battery performs generalized principal components analysis and inference tests on a data matrix.

If the expected time to compute the results (based on test.iters) exceeds 1 minute, you will be asked (via command line) if you want to continue.

Value

Returns two lists ($Fixed.Data and $Inference.Data). For $Fixed.Data, see epGPCA, corePCA for details on the descriptive (fixed-effects) results.

$Inference.Data returns:

- components: Permutation tests of components. p-values ($p.vals) and distributions of eigenvalues ($eigs.perm) for each component

Author(s)

Derek Beaton and Hervé Abdi.

See Also

epGPCA, epPCA, epPCA.inference.battery

Examples

```r
#this is for ExPosition's iris data
data(ep.iris)
data<-ep.iris$data
design<-ep.iris$design
gpca.iris.res <- epGPCA.inference.battery(data,DESIGN=design,make_design_nominal=FALSE)
```

epMCA.inference.battery

epMCA.inference.battery: Inference tests for Multiple Correspondence Analysis (CA) via InPosition.

Description

Multiple Correspondence Analysis (CA) and a battery of inference tests via InPosition. The battery includes permutation and bootstrap tests.
Usage

epMCA.inference.battery(DATA, make_data_nominal = TRUE, DESIGN = NULL, 
make_design_nominal = TRUE, masses = NULL, weights = NULL, 
hellinger = FALSE, symmetric = TRUE, correction = c("b"), 
graphs = TRUE, k = 0, 
test.iters = 100, constrained = FALSE, critical.value = 2)

Arguments

DATA original data to perform a MCA on. This data can be in original formatting 
(qualitative levels) or in dummy-coded variables.

make_data_nominal  
a boolean. If TRUE (default), DATA is recoded as a dummy-coded matrix. If 
FALSE, DATA is a dummy-coded matrix.

DESIGN a design matrix to indicate if rows belong to groups.

make_design_nominal  
a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and 
will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.

masses a diagonal matrix or column-vector of masses for the row items.

weights a diagonal matrix or column-vector of weights for the column it

hellinger a boolean. If FALSE (default), Chi-square distance will be used. If TRUE, 
Hellinger distance will be used.

symmetric a boolean. If TRUE symmetric factor scores for rows.

correction which corrections should be applied? "b" = Benzécri correction, "bg" = Greenacre 
adjustment to Benzécri correction.

graphs a boolean. If TRUE (default), graphs and plots are provided (via epGraphs)

k number of components to return.

test.iters number of iterations

constrained a boolean. If a DESIGN matrix is used, this will constrain bootstrap resampling 
to be within groups.

critical.value numeric. A value, analogous to a z- or t-score to be used to determine signifi-
cance (via bootstrap ratio).

Details

epMCA.inference.battery performs multiple correspondence analysis and inference tests on a 
data matrix.

If the expected time to compute the results (based on test.iters) exceeds 1 minute, you will be asked (via command line) if you want to continue.
epPCA.inference.battery

Value

Returns two lists ($Fixed.Data and $Inference.Data). For $Fixed.Data, see epMCA, coreCA for details on the descriptive (fixed-effects) results.

$Inference.Data returns:

- **components**: Permutation tests of components, p-values ($p.vals) and distributions of eigenvalues ($eigs.perm) for each component.
- **fj.boots**: Bootstrap tests of measures (columns). See boot.ratio.test output details.
- **omni**: Permutation tests of components. p-values ($p.val) and distributions of total inertia ($inertia.perm). This is only useful if corrections are performed. Total inertia is constant for permutation with no corrections in MCA.

Author(s)

Derek Beaton, Joseph Dunlop, and Hervé Abdi.

See Also

epMCA, epCA, epCA.inference.battery

Examples

```r
data(mca.wine)
mca.wine.res <- epMCA.inference.battery(mca.wine$data)
```

Description

Principal Component Analysis (PCA) and a battery of inference tests via InPosition. The battery includes permutation and bootstrap tests.

Usage

```r
eppca.inference.battery(DATA, scale = TRUE, center = TRUE, DESIGN = NULL, make_design_nominal = TRUE, graphs = TRUE, k = 0, test.iters = 100, constrained = FALSE, critical.value = 2)
```
Arguments

- **DATA**: original data to perform a PCA on.
- **scale**: a boolean, vector, or string. See `expo.scale` for details.
- **center**: a boolean, vector, or string. See `expo.scale` for details.
- **DESIGN**: a design matrix to indicate if rows belong to groups.
- **make_design_nominal**: a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.
- **graphs**: a boolean. If TRUE (default), graphs and plots are provided (via `epGraphs`)
- **k**: number of components to return.
- **test.iters**: number of iterations
- **constrained**: a boolean. If a DESIGN matrix is used, this will constrain bootstrap resampling to be within groups.
- **critical.value**: numeric. A value, analogous to a z- or t-score to be used to determine significance (via bootstrap ratio).

Details

epPCA.inference.battery performs principal components analysis and inference tests on a data matrix.

If the expected time to compute the results (based on `test.iters`) exceeds 1 minute, you will be asked (via command line) if you want to continue.

Value

Returns two lists (`$Fixed.Data` and `$Inference.Data`). For `$Fixed.Data`, see `epPCA, corePCA` for details on the descriptive (fixed-effects) results.

- **$Inference.Data** returns:
  - **components**: Permutation tests of components. p-values (`$p.vals`) and distributions of eigenvalues (`$eigs.perm`) for each component
  - **fj.boots**: Bootstrap tests of measures (columns). See `boot.ratio.test` output details.

Author(s)

Derek Beaton and Hervé Abdi.

See Also

- `epPCA, epGPCA, epGPCA.inference.battery`

Examples

```r
data(words)
pca.words.res <- epPCA.inference.battery(words$data)
```
inGraphs

inGraphs: InPosition plotting function

Description

InPosition plotting function which is an interface to prettyGraphs.

Usage

inGraphs(res, DESIGN = NULL, x_axis = NULL, y_axis = NULL, inference.info = NULL,
        color.by.boots = TRUE, boot.cols = c("plum4", "darkseagreen", "firebrick3"),
        fi.col = NULL, fi.pch = NULL, fj.col = NULL, fj.pch = NULL, col.offset = NULL,
        constraints = NULL, xlab = NULL, ylab = NULL, main = NULL, bootstrapBars = TRUE,
        correlationPlotter = TRUE, biplots = FALSE)

Arguments

res results from InPosition or ExPosition. If results are from ExPosition, inference.info
        must be included.
        DESIGN A design matrix to apply colors (by pallete selection) to row items
x_axis which component should be on the x axis?
y_axis which component should be on the y axis?
inference.info Inference data as output by InPosition (of class inpoOutput).
color.by.boots a boolean. If TRUE, items are colored by bootstrap ratio test. Items larger than
        critical.value are colored 'plum4' on the horizontal component, 'darkseagreen'
        on the vertical component, or 'firebrick3' if the item is significant on
        both components (to be visualized). If FALSE, the color of the items will
        be used.
        boot.cols vector of colors: c(horizontal component color, vertical component color, color when item
        fi.col A matrix of colors for the row items. If NULL, colors will be selected.
        fi.pch A matrix of pch values for the row items. If NULL, pch values are all 21.
        fj.col A matrix of colors for the column items. If NULL, colors will be selected.
        fj.pch A matrix of pch values for the column items. If NULL, pch values are all 21.
        col.offset A numeric offset value. Is passed to createColorVectorsByDesign.
        constraints Plot constraints as returned from prettyPlot. If NULL, constraints are
        selected.
        xlab x axis label
        ylab y axis label
        main main label for the graph window
        bootstrapBars a boolean. If TRUE (default), bootstrap ratio bar plots will be created.
correlationPlotter

a boolean. If TRUE (default), a correlation circle plot will be created. Applies to PCA family of methods (CA is excluded for now).

biplots

a boolean. If FALSE (default), separate plots are made for row items ($fi$) and column items ($fj$). If TRUE, row ($fi$) and column ($fj$) items will be on the same plot.

Value

Currently, nothing is returned. This function, for now, works as a visualizer for inference tests. Colors and constraints come from the descriptive (fixed effects) analysis.

Author(s)

Derek Beaton

See Also

epGraphs

Examples

data(ep.iris)
data<-ep.iris$data
design<-ep.iris$design
g pca.iris.res <- epGPCA.inference.battery(data,DESIGN=design,make_design_nominal=FALSE)
inGraphs(g pca.iris.res,y_axis=3)

print.epCA.inference.battery

Print Correspondence Analysis (CA) Inference results

Description

Print Correspondence Analysis (CA) Inference results.

Usage

```r
## S3 method for class 'epCA.inference.battery'
print(x, ...)
```

Arguments

- **x**: an list that contains items to make into the epCA.inference.battery class.
- **...**: inherited/passsed arguments for S3 print method(s).

Author(s)

Derek Beaton and Cherise Chin-Fatt
**print.epGPCA.inference.battery**

*Print Generalized Principal Components Analysis (GPCA) Inference results*

---

**Description**

Print Generalized Principal Components Analysis (GPCA) Inference results.

**Usage**

```r
## S3 method for class 'epGPCA.inference.battery'
print(x, ...)
```

**Arguments**

- `x` an list that contains items to make into the epGPCA.inference.battery class.
- `...` inherited/passed arguments for S3 print method(s).

**Author(s)**

Derek Beaton and Cherise Chin-Fatt

---

**print.epMCA.inference.battery**

*Print Multiple Correspondence Analysis (MCA) Inference results*

---

**Description**

Print Multiple Correspondence Analysis (MCA) Inference results.

**Usage**

```r
## S3 method for class 'epMCA.inference.battery'
print(x, ...)
```

**Arguments**

- `x` an list that contains items to make into the epMCA.inference.battery class.
- `...` inherited/passed arguments for S3 print method(s).

**Author(s)**

Derek Beaton and Cherise Chin-Fatt
print.epPCA.inference.battery

*Print Principal Components Analysis (PCA) Inference results*

**Description**

Print Principal Components Analysis (PCA) Inference results.

**Usage**

```r
## S3 method for class 'epPCA.inference.battery'
print(x,...)
```

**Arguments**

- `x` an list that contains items to make into the epPCA.inference.battery class.
- `...` inherited/passed arguments for S3 print method(s).

**Author(s)**

Derek Beaton and Cherise Chin-Fatt

---

print.inpoboot

*Print results from InPosition Bootstraps*

**Description**

Print bootstrap results from the InPosition.

**Usage**

```r
## S3 method for class 'inpoboot'
print(x,...)
```

**Arguments**

- `x` an list that contains items to make into the inpoboot class.
- `...` inherited/passed arguments for S3 print method(s).

**Author(s)**

Derek Beaton and Cherise Chin-Fatt
print.inpocomponents

**print.inpocomponents**  
*Print results from InPosition Components Permutation Test*

---

**Description**  
Print Components permutation test results from the inposition.

**Usage**  
```r  
## S3 method for class 'inpocomponents'  
print(x,...)  
```

**Arguments**  
- `x`: an list that contains items to make into the inpocomponents class.  
- `...`: inherited/passed arguments for S3 print method(s).

**Author(s)**  
Derek Beaton and Cherise Chin-Fatt

---

print.inpoboottests

**print.inpoboottests**  
*Print results from InPosition Bootstrap Ratio Tests*

---

**Description**  
Print bootstrap ratio tests results from the InPosition.

**Usage**  
```r  
## S3 method for class 'inpoboottests'  
print(x,...)  
```

**Arguments**  
- `x`: an list that contains items to make into the inpoboottests class.  
- `...`: inherited/passed arguments for S3 print method(s).

**Author(s)**  
Derek Beaton and Cherise Chin-Fatt
Description

Print Omnibus permutation test results from the inposition.

Usage

```r
## S3 method for class 'inpoOmni'
print(x,...)
```

Arguments

- `x`: an list that contains items to make into the inpoOmni class.
- `...`: inherited/passed arguments for S3 print method(s).

Author(s)

Derek Beaton and Cherise Chin-Fatt

See Also

eppcaNinferenceNbattery, inGraphs
rebuildMCAtable: rebuild categorical table from the disjunctive table.

Description

rebuildMCAtable takes the disjunctive table used in MCA and rebuilds a categorical form of it. This function is used for permutation tests when only a disjunctive table is available.

Usage

rebuildMCAtable(data)

Arguments

- **DATA**  
  Disjunctive coded data table

Value

A categorical data table is returned. It has the same structure as the disjunctive table in a format that can be permuted.

Author(s)

Derek Beaton
Index

*Topic **bootstrap**
  boot.compute.fj, 3
  boot.ratio.test, 4
  boot.samples, 6
  contingency.data.break, 8
  epCA.inference.battery, 9
  epGPCA.inference.battery, 11
  epMCA.inference.battery, 12
  epPCA.inference.battery, 14
  inGraphs, 16

*Topic **correspondence analysis**
  caChiTest, 7

*Topic **graphs**
  inGraphs, 16

*Topic **inference**
  boot.compute.fj, 3
  boot.ratio.test, 4
  caChiTest, 7

*Topic **misc**
  caChiTest, 7
  inGraphs, 16
  rebuildMCATable, 22

*Topic **multivariate**
  boot.compute.fj, 3
  boot.ratio.test, 4
  caChiTest, 7
  epCA.inference.battery, 9
  epGPCA.inference.battery, 11
  epMCA.inference.battery, 12
  epPCA.inference.battery, 14
  inGraphs, 16
  InPosition-package, 2
  rebuildMCATable, 22

*Topic **package**
  InPosition-package, 2

*Topic **permutation**
  contingency.data.break, 8
  epCA.inference.battery, 9
  epGPCA.inference.battery, 11
  epMCA.inference.battery, 12
  epPCA.inference.battery, 14
  inGraphs, 16
  print.epCA.inference.battery, 17
  print.epGPCA.inference.battery, 18
  print.epMCA.inference.battery, 18
  print.epPCA.inference.battery, 19
  print.inpoBoot, 19
  print.inpoBootTests, 20
  print.inpoComponents, 20
  print.inpoOmni, 21
  print.inpoOutput, 21

array, 5

boot.compute.fj, 3, 5, 6
boot.ratio.test, 4, 6, 10, 12, 14, 15
boot.samples, 6

cachiTest, 7, 10
contingency.data.break, 8
continueResampling, 9
coreCA, 10, 14
corePCA, 12, 15
createColorVectorsByDesign, 16

epCA, 8, 10, 14
epCA.inference.battery, 3, 8, 9, 14
epGPCA, 12, 15
epGPCA.inference.battery, 3, 11, 15
epGraphs, 10, 11, 13, 15, 17
epMCA, 3, 10, 14
epMCA.inference.battery, 3, 10, 12
epPCA, 3, 12, 15
epPCA.inference.battery, 3, 12, 14, 21
expo.scale, 11, 15
ExPosition, 2, 3

inGraphs, 3, 16, 21
InPosition(InPosition-package), 2
InPosition-package, 2
prettyGraphs, 16
prettyPlot, 16
print.epCA.inference.battery, 17
print.epGPCA.inference.battery, 18
print.epMCA.inference.battery, 18
print.epPCA.inference.battery, 19
print.inpoBoot, 19
print.inpoBootTests, 20
print.inpoComponents, 20
print.inpoOmmi, 21
print.inpoOutput, 21
rebuildMCAtable, 22
supplementaryCols, 4