Package 'lsbclust'

January 5, 2016

Type Package

Title Least-Squares Bilinear Clustering for Three-Way Data

Version 1.0.4

Date 2016-01-05

Author Pieter Schoonees [aut, cre], Patrick Groenen [ctb]

Maintainer Pieter Schoonees <schoonees@gmail.com>

Description Functions for performing least-squares bilinear clustering of three-way data. The method uses the bilinear decomposition (or biadditive model) to model two-way matrix slices while clustering over the third way. Up to four different types of clusters are included, one for each term of the bilinear decomposition. In this way, matrices are clustered simultaneously on (a subset of) their overall means, row margins, column margins and row-column interactions. The orthogonality of the bilinear model results in separability of the joint clustering problem into four separate ones. Three of these subproblems are specific k-means problems, while a special algorithm is implemented for the interactions. Plotting methods are provided, including biplots for the low-rank approximations of the interactions.

License GPL (>= 2)

Depends R (>= 3.2.2), stats, parallel, ggplot2

Imports plyr, clue, grid, gridExtra, reshape2, Rcpp, graphics, methods

LinkingTo Rcpp

LazyData yes

LazyLoad yes

ByteCompile yes

BuildResaveData best

Suggests testthat

NeedsCompilation yes

RoxygenNote 5.0.1

Repository CRAN

Date/Publication 2016-01-05 14:18:01

R topics documented:

`lsbclust-package` *Least Squares Latent Class Matrix Factorization*

Description

Funtions for least squares latent class matrix factorizations.

Author(s)

Pieter C. Schoonees [aut, cre], Patrick J.F. Groenen [aut]

References

Van Rosmalen, J., Van Herk, H., & Groenen, P. J. F. (2010). Identifying response styles: A latent-class bilinear multinomial logit model. *Journal of Marketing Research*, 47(1), 157-172.

bicomp *Bilinear Decomposition of a Matrix*

Description

Decomposes a matrix into an overall mean matrix, row margins matrix, column margins matrix and an interaction matrix, depending on `delta`.

Usage

```
bicomp(x, delta = c(1, 1, 1, 1), which = 0L:4L)
```

Arguments

`x`	A matrix to be decomposed.
`delta`	A vector of length four with 0/1 entries which controls the type of decomposition made.
`which`	A vector giving the elements to return, with 0 = original data, 1 = overall means, 2 = row means, 3 = column means and 4 = interactions.

Value

An object of class `bicomp`, possible also inheriting from class `data.frame`, which is either a named list with the required components, or a single matrix if a single component is requested. An additional attribute `return_type` gives information on the type of matrices returned.

ClustMeans *C++ Function for Cluster Means*

Description

This function calculates the cluster means in vectorized form based on the current value of the clustering vector.

Usage

```
ClustMeans(nclust, start, data)
```

Arguments

`nclust`	The number of clusters.
`start`	The current clustering vector.
`data`	The concatenated data, with J * K rows and N columns

Value

A numeric matrix with `nclust` rows and `J*K` columns.

cl_class_ids.int.lsbclust *S3 Methods for Integration into* **clue** *Framework*

Description

These methods integrates the class `int.lsbclust` into the framework set out in package **clue**.

Usage

```
## S3 method for class 'int.lsbclust'
cl_class_ids(x)

## S3 method for class 'int.lsbclust'
is.cl_partition(x)

## S3 method for class 'int.lsbclust'
is.cl_hard_partition(x)
```

Arguments

`x`	An object of class `int.lsclust`

cmat *Centring Matrix*

Description

A utility function for calculating centring matrices.

Usage

```
cmat(k)
```

Arguments

`k`	An integer determining the dimensions of the centring matrix.

dcars *Dutch Cars Data*

Description

This data set relates to 187 Dutch households rating 10 automobile manufacturers according to 8 variables (original Dutch terms in parentheses): price (prijsniveau), design (vormgeving), safety (veiligheid), operating cost (gebruikskosten),) sportiness (sportiviteit), size (modelgrootte), reliability (betrouwbaarheid) and feautures (uitrusting). A rating scale from 1 to 10 was used.

Usage

```
dcars
```

Format

A three-way array with cars in the first dimension, variables in the second and consumers in the third dimension.

The items and labels for the endpoints of the scales are (original Dutch labels in parentheses):

Affordability A rating from 1 = Expensive (duur) to 10 = Cheap (goedkoop)

Attractiveness A rating from 1 = Ugly (lelijk) to 10 = Beautiful (mooi)

Safety A rating from 1 = Bad (slecht) to 10 = Good (goed)

OperatingCost A rating from 1 = Low (laag) to 10 = High (hoog)

Sportiness A rating from 1 = Slow (langzaam) to 10 = Fast (snel)

Size A rating from 1 = Large (groot) to 10 = Small (klein)

Reliability A rating from 1 = Bad (slecht) to 10 = Good (goed)

Features A rating from 1 = Simple (eenvoudig) to 10 = Luxurious (luxe)

Details

The original sample consisted of 188 households. However, one of these households (code 87845) was discarded because it appears that they used a rating scale from 0 to 10 instead of from 1 to 10. Note that all rating scales has been reversed so that higher scores are better for most items. The exceptions are OperatingCost and Size, where larger values mean higher costs and smaller cars respectively.

Source

Tammo Bijmolt, Michel van de Velden

Examples

```
data("dcars")
set.seed(5448)
m <- lsbclust(data = dcars, delta = c(1, 1, 1, 1), nclust = c(5, 3, 6, 8), nstart = 5,
              nstart.kmeans = 10, parallelize = FALSE, fixed = "columns")
```

genproc *Generalized Procrustes Rotation*

Description

This function finds K orthogonal rotation matrices so that the rotated versions of the input configurations match each other optimally in the least-squares sense. The algorithm depends on the starting values for the rotation matrices. At present identity matrices are used as starting values. Only rotations / reflections are considered – no scaling or translation factors are included.

Usage

```
genproc(configs, maxit = 50L, reltol = 1e-06, random = FALSE)
```

Arguments

configs	A list of original configuration matrices
maxit	The maximum number of iterations allowed
reltol	The relative error tolerance for determining numeric convergence.
random	Logical indicating whether or not to use random starts (only applicable when the dimensionality is two).

References

Gower, J. C., & Dijksterhuis, G. B. (2004). Procrustes problems (Vol. 3). Oxford: Oxford University Press.

indarr *Create Array of Indicator Matrices*

Description

This function takes a `matrix` or `data.frame` and the number of rating categories `maxcat` and produces a three-way array of `m` by `maxcat` indicator matrices, one for each of the `n` rows. The input `x` must be a `matrix` or `data.frame` of dimensions `n` by `m` which contains the ratings on a scale of 1 to `maxcat` for `m` items. Note that missing values (`NA`'s) will not appear in the columns.

Usage

```
indarr(x, maxcat, na.add = TRUE)
```

Arguments

x	a matrix of data.frame
maxcat	an integer indicating the maximum of the rating scale (which is assumed to start with 1)
na.add	logical indicating whether to add a designated category for missings or not. Defaults to TRUE.

Value

A list of rating by item indicator matrices.

Author(s)

Pieter C. Schoonees

Examples

```
data("lov")
arr <- indarr(lov[1:10, 1:9], maxcat = 9)
str(arr)
```

---

int.lsbclust *Interaction Clustering in Least Squares Bilinear Clustering*

Description

This function implements the interaction clustering part of the Least Squares Bilinear Clustering method of Schoonees, Groenen and Van de Velden (2014).

Usage

```
int.lsbclust(data, margin = 3L, delta, nclust, ndim = 2, fixed = c("none",
  "rows", "columns"), nstart = 50, starts = NULL, alpha = 0.5,
  parallelize = FALSE, maxit = 100, verbose = 1, method = "diag")
```

Arguments

data	A three-way array representing the data.
margin	An integer giving the single subscript of data over which the clustering will be applied.
delta	A four-element binary vector (logical or numeric) indicating which sum-to-zero constraints must be enforced.
nclust	An integer giving the desired number of clusters. If it is a vector, the algorithm will be run for each element.
ndim	The required rank for the approximation of the interactions (a scalar).

fixed	One of "none", "rows" or "columns" indicating whether to fix neither sets of coordinates, or whether to fix the row or column coordinates across clusters respectively. If a vector is supplied, only the first element will be used.
nstart	The number of random starts to use.
starts	A list containing starting configurations for the cluster membership vector. If not supplied, random initializations will be generated.
alpha	Numeric value in [0, 1] which determines how the singular values are distributed between rows and columns.
parallelize	Logical indicating whether to parallelize over different starts or not.
maxit	The maximum number of iterations allowed.
verbose	Integer controlling the amount of information printed: 0 = no information, 1 = Information on random starts and progress, and 2 = information is printed after each iteration for the interaction clustering.
method	The method for calculating cluster agreement across random starts, passed on to cl_agreement. None is calculated when set to NULL.

Value

An object of class int.lsb

Examples

```
data("supermarkets")
out <- int.lsbclust(data = supermarkets, margin = 3, delta = c(1,1,0,0), nclust = 4, ndim = 2,
           fixed = "rows", nstart = 1, alpha = 0)
```

KMeansW *C++ Function for Weighted K-Means*

Description

This function does a weighted K-means clustering.

Usage

```
ComputeMeans(cm, data, weight, nclust)

AssignCluster(data, weight, M, nclust)

KMeansW(nclust, start, data, weight, eps = 1e-08, IterMax = 100L)
```

Arguments

cm	Numeric vector of class indicators.
data	The concatenated data, with N rows and M columns. Currently, the columns are clustered.
weight	The vector of length nrows(data) with weights with nonnegative elements.
nclust	The number of clusters.
M	Matrix of cluster means.
start	The current cluster membership vector.
eps	Numerical absolute convergence criteria for the K-means.
IterMax	Integer giving the maximum number of iterations allowed for the K-means.

Value

A list with the folowing values.

centers	the nclust by M matrix centers of cluster means.
cluster	vector of length N with cluster memberships.
loss	vector of length IterMax with the first entries containing the loss.
iterations	the number of iterations used (corresponding to the number of nonzero entries in loss)

Examples

```
set.seed(1)
clustmem <- sample.int(n = 10, size = 100, replace = TRUE)
mat <- rbind(matrix(rnorm(30*4, mean = 3), nrow = 30),
             matrix(rnorm(30*4, mean = -2), nrow = 30),
             matrix(rnorm(40*4, mean = 0), nrow = 40))
wt <- runif(100)
testMeans <- lsbclust:::ComputeMeans(cm = clustmem, data = mat, weight = wt, nclust = 3)
testK <- lsbclust:::KMeansW(start = clustmem, data = mat, weight = wt, nclust = 3)
```

LossMat *C++ Function for Interaction Loss Function*

Description

This function calculates the loss function for the interaction clustering for all data slices and clusters means. The inputs are numeric matrices.

Arguments

x	The data matrix, with the N slices strung out as vectors in the columns.
y	The matrix of cluster means, with each mean represented by a row.

Value

A numeric matrix with `nclust` rows and `N` columns.

`lov` *List-of-values Data Set*

Description

This is the list-of-values data set used in Van Rosmalen, Van Herk & Groenen (2010). Column names and factor labels differ slightly from that paper. Missing values are encoded as `NA` as usual. The first nine columns are items answered on a nine-point rating scale, with rating 1 representing 'very important' and category 9 'not important at all'. The respondents were asked how important each of these items are as a guiding principle in their lives.

Usage

```
data("lov")
```

Format

A data frame with 4514 observations on the following 12 variables.

Belonging a numeric vector; 'a sense of belonging'

Excitement a numeric vector

Relationships a numeric vector; 'warm relationships with others'

Self-fulfilment a numeric vector

Respected a numeric vector; 'being well-respected'

Enjoyment a numeric vector; 'fun and enjoyment'

Security a numeric vector

Self-respect a numeric vector

Accomplishment a numeric vector; 'a sense of accomplishment'

Country a factor with levels `Britain`, `France`, `Germany`, `Italy` and `Spain`

Education a factor with levels `Low` and `High`

Age a factor with levels `-25`, `25-39`, `40-54` and `55+`

Source

Joost van Rosmalen

References

Van Rosmalen, J., Van Herk, H., & Groenen, P. J. (2010). Identifying response styles: A latent-class bilinear multinomial logit model. *Journal of Marketing Research*, 47(1), 157-172.

Examples

```
data("lov")

## Construct array
lovarr <- indarr(lov[, 1:9], maxcat = 9)

## Run analysis
set.seed(13841)
fit <- lsbclust(data = lovarr, margin = 3, delta = c(0, 1, 0, 0), nclust = c(NA, 11, NA, 5),
                fixed = "rows", nstart = 1, iter.max = 50, nstart.kmeans = 10)
```

`lsbclust` *Least-squares Bilinear Clustering of Three-way Data*

Description

This function clusters along one way of a three-way array (as specified by `margin`) while decomposing along the other two dimensions. Four types of clusterings are allowed based on the respective two-way slices of the array: on the overall means, row margins, column margins and the interactions between rows and columns. Which clusterings can be fit is determined by the vector `delta`, with four binary elements. All orthogonal models are fitted. The nonorthogonal case `delta = (1, 1, 0, 0)` returns an error. See the reference for further details.

Usage

```
lsbclust(data, margin = 3L, delta = c(1, 1, 1, 1), nclust, ndim = 2,
  fixed = c("none", "rows", "columns"), nstart = 20, starts = NULL,
  nstart.kmeans = 500, alpha = 0.5, parallelize = FALSE, maxit = 100,
  verbose = 1, method = "diag", type = NULL, sep.nclust = TRUE, ...)
```

Arguments

`data`	A three-way array representing the data.
`margin`	An integer giving the single subscript of `data` over which the clustering will be applied.
`delta`	A four-element binary vector (logical or numeric) indicating which sum-to-zero constraints must be enforced.
`nclust`	A vector of length four giving the number of clusters for the overall mean, the row margins, the column margins and the interactions (in that order) respectively. Alternatively, a vector of length one, in which case all components will have the same number of clusters.
`ndim`	The required rank for the approximation of the interactions (a scalar).
`fixed`	One of `"none"`, `"rows"` or `"columns"` indicating whether to fix neither sets of coordinates, or whether to fix the row or column coordinates across clusters respectively. If a vector is supplied, only the first element will be used (passed to `int.lsbclust`).

nstart	The number of random starts to use for the interaction clustering.
starts	A list containing starting configurations for the cluster membership vector. If not supplied, random initializations will be generated (passed to int.lsbclust).
nstart.kmeans	The number of random starts to use in kmeans.
alpha	Numeric value in [0, 1] which determines how the singular values are distributed between rows and columns (passed to int.lsbclust).
parallelize	Logical indicating whether to parallelize over different starts or not (passed to int.lsbclust).
maxit	The maximum number of iterations allowed in the interaction clustering.
verbose	Integer controlling the amount of information printed: 0 = no information, 1 = Information on random starts and progress, and 2 = information is printed after each iteration for the interaction clustering.
method	The method for calculating cluster agreement across random starts, passed on to cl_agreement (passed to int.lsbclust).
type	One of "rows", "columns" or "overall" (or a unique abbreviation of one of these) indicating whether clustering should be done on row margins, column margins or the overall means of the two-way slices respectively. If more than one opion are supplied, the algorithm is run for all (unique) options supplied (passed to orc.lsbclust). This is an optional argument.
sep.nclust	Logical indicating how nclust should be used across different type's. If sep.nclust is TRUE, nclust is recycled so that each type can have a different number of clusters. If sep.nclust is FALSE, the same vector nclust is used for all type's.
...	Additional arguments passed to kmeans.

Value

Returns an object of S3 class lsbclust which has slots:

overall	Object of class ovl.kmeans for the overall means clustering
rows	Object of class row.kmeans for the row means clustering
columns	Object of class col.kmeans for the column means clustering
interactions	Object of class int.lsbclust for the interaction clustering
call	The function call used to create the object
delta	The value of delta in the fit
df	Breakdown of the degrees-of-freedom across the different subproblems
loss	Breakdown of the loss across subproblems
time	Time taken in seconds to calculate the solution
cluster	Matrix of cluster membership per observation for all cluster types

References

Schoonees, P.C., Groenen, P.J.F., Van de Velden, M. Least-squares Bilinear Clustering of Three-way Data. Econometric Institute Report, EI2014-23.

See Also

int.lsbclust, orc.lsbclust

---

orc.lsbclust *K-means on the Overall Mean, Row Margins or Column Margins*

Description

This function conducts k-means on the overall mean, the row margins or column margins of a set of N matrices. These matrices are two-way slices of a three-dimensional array.

Usage

```
orc.lsbclust(data, margin = 3L, delta, nclust, sep.nclust = TRUE,
  type = NULL, verbose = 1, ...)
```

Arguments

data	A three-way array representing the data.
margin	An integer giving the single subscript of data over which the clustering will be applied.
delta	A four-element binary vector (logical or numeric) indicating which sum-to-zero constraints must be enforced.
nclust	An integer giving the desired number of clusters. In case type specifies more than one method, nclust can be a vector containing the number of clusters to be determined for each type of cluster, and in the correct order as determined by type (after matching the arguments). If type is of length greater than one and nclust is of length one, the behaviour is governed by sep.nclust.
sep.nclust	Logical indicating how nclust should be used across different type's. If sep.nclust is TRUE, nclust is recycled so that each type can have a different number of clusters. If sep.nclust is FALSE, the same vector nclust is used for all type's.
type	One of "overall", "rows" or "columns" (or a unique abbreviation of one of these) indicating whether clustering should be done on row margins, column margins or the overall means of the two-way slices respectively. If more than one opion are supplied, the algorithm is run for all (unique) options supplied.
verbose	Integer controlling the amount of information printed: 0 = no information, 1 = Information on random starts and progress, and 2 = information is printed after each iteration for the interaction clustering.
...	Additional arguments passed to kmeans.

Value

A list containing a subset of the classes row.kmeans, col.kmeans and ovl.kmeans which are specific versions of class kmeans. In case type is a vector, a list is returned containing the results for each of the (unique) elements of type, with the same classes as before. See kmeans for an overview of the structure of these objects.

See Also

kmeans

---

plot.bicomp *Plot a* bicomp *Object*

Description

Plot method for an object of class bicomp (see bicomp).

Usage

```
## S3 method for class 'bicomp'
plot(x, which = 0L:4L, arrange = TRUE, col = c("red4",
  "beige", "blue4"), strip.legend = TRUE, add.titles = FALSE, ...)
```

Arguments

x	An object of class bicomp.
which	A numeric vector indicating which matrices to plot, with 0 = original data, 1 = overall means, 2 = row means, 3 = column means and 4 = interactions.
arrange	Logical indicating whether the arrange the plots side-by-side via grid.arrange or not.
col	A character vector of length three giving the parameters low, mid and high for scale_fill_gradient2.
strip.legend	Logical indicating whether to strip the legend off the plot or not.
add.titles	Logical indicating whether to add titles to the plots or not.
...	Additional arguments to theme.

---

plot.col.kmeans *Plot method for class 'col.kmeans'*

Description

Simple plot method for object of class 'col.kmeans' as output by orc.lsbclust.

Usage

```
## S3 method for class 'col.kmeans'
plot(x, which = 1L, ...)
```

Arguments

x	An object of class col.kmeans
which	Which type of plot to produce (only 3 types are implemented).
...	additional arguments passed to theme.

Author(s)

Pieter C. Schoonees

Examples

```
data("dcars")
m <- orc.lsbclust(data = dcars, margin = 3, delta = c(1,1,1,1), nclust = 5, type = "columns")
plot(m)
```

---

plot.int.lsbclust *Plot Method for Class 'int.lsbclust'*

Description

Two-dimensional plot method for object of class 'int.lsbclust' as output by int.lsbclust.

Usage

```
## S3 method for class 'int.lsbclust'
plot(x, which = seq_len(nclust),
  plot.type = c("biplots", "means", "estimates"), segments = NULL,
  biplot.axes = TRUE, nmarkers = 5, alpha = NULL, check.alpha = TRUE,
  fix.alpha = FALSE, probs = 0, arrange = FALSE, fix.limits = TRUE,
  limit.exp = 1.05, lambda.scale = TRUE, procrustes.rotation = x$fixed ==
  "none", fix.lambda = FALSE, labs.grey = TRUE, label.0 = FALSE,
  tick.length = 0.0075 * diff(lims), axis.col = "grey60", label.size = 3,
  axis.size = 0.25, axis.title.size = 4, draw.axis = NULL,
  points.col = list(rows = "red", columns = "blue2"),
  offset.tick.labels = 3.5, offset.axis.title = list(rows = 0.015 *
  max(nchar(rnms)), columns = 0.015 * max(nchar(cnms))),
  axis.arrow = grid::arrow(angle = 20, length = grid::unit(0.0175, "npc")),
  ...)
```

Arguments

x	An object of class int.lsbclust.
which	A vector indicating which item segments to plot.
plot.type	Character string giving the type of plots to produce: either "biplots" for the biplots approximating the cluster means, "means" for level plots of the cluster means themselves or "estimates" for level plots of the low-rank approximations of the cluster means (as represented in the biplots).

segments — A logical vector with two elements, indicating whether the rows and columns should be plotted as line segments or not.

biplot.axes — A logical indicating whether to plot calibrated biplot axes for the line segments indicated in segments or not.

nmarkers — Either a single integer giving the number of desired markers per biplot axis for all axes, or a named list. This is passed as the argument n to pretty. See Details for information on the list option.

alpha — Numeric value in [0, 1] which determines how the singular values are distributed between rows and columns. It will trigger a recomputation of the updates if it does not correspond to the value used when fitting the model.

check.alpha — Logical indicating whether to look for a better alpha. This is only used when alpha = NULL is used.

fix.alpha — Logical indicating whether to fix alpha across all clusters or not when fixed == "none".

probs — Argument passed to quantile to determine the alpha value. The corresponding quantile of the distances of all points in the biplots to the origin will be used to determine alpha in case check.alpha = TRUE.

arrange — Logical indicating whether to arrange the plots side-by-side via grid.arrange or not.

fix.limits — Logical indicating whether biplot x- and y-limits must be fixed across clusters or not. Note that this is automatically set to TRUE when fixed == "rows" or fixed == "columns". When limits are fixed, the axis calibrations are also turned off.

limit.exp — A numeric expansion factor applied multiplicatively to the plot limits, but only when fixed equals "rows" or "columns".

lambda.scale — Logical indicating whether to apply lambda scaling to the coordinates or not. If true, the scaling is done such that the average squared distance to the origin is equal for the row and column coordinates.

procrustes.rotation — Logical indicating whether to do Procrustes rotations so that the location of the axes indicated as segments (see argument segments) are similar across configurations.

fix.lambda — Logical indicating whether to fix lambda across all clusters or not.

labs.grey — Logical indicating whether to apply greying to the text labels are well.

label.0 — Logical indicating whether to label the origin or not.

tick.length — The required tick length as a unit object. It defaults to a propoprtion of the width of the plot region (through lazy evaluation).

axis.col — The colour of the biplot axes.

label.size — The size of the labels for the markers on the biplot axes.

axis.size — Line size for biplot axes.

axis.title.size — Size of biplot axis titles.

`draw.axis` A list with up to two components which must be named `"rows"` and `"columns"`. Each element contains a vector indicating which biplot axes should be drawn. The vectors can be character vectors containing the names of the axes to be drawn, numeric vectors containing indices indicating which axes to draw, or logical vectors indicating which biplot axes to draw. In case of the default value `NULL`, the elements of `segments` are used for the `"rows"` and `"columns"` entries.

`points.col` A named list containing the colours to use for plotting the sets of points. The elements `"rows"` and `"columns"` contain vectors giving the colours for the points. Single element vectors are recycled across the different points, otherwise the vectors must be of the appropriate length.

`offset.tick.labels`
A numeric value giving the offset factor of the biplot axis marker labels from their respective tick marks. Higher (lower) values lead to labels being further from (nearer to) their respective tick marks.

`offset.axis.title`
A names list of (up to) two numeric values giving the fixed length offset of the biplot axis title label from the end of the axis segment. The two elements must have names `"rows"` and code"columns".

`axis.arrow` An `arrow` object to be used for the endpoints of biplot axis segment lines. This is passed to `geom_segment`.

`...` Additional arguments passed to `theme`.

Details

In case `nmarkers` is a list, it can have up to two elements. These are required to be named `"rows"` and/or `"columns"`, otherwise an error will be thrown. The elements of the list contains either single numeric values each or numeric vectors of the appropriate lengths indicating the `n` argument passed to `pretty`.

---

plot.lsbclust *Plot method for class 'lsbclust'*

Description

This plot method simply plots each of the components in the list of class `lsbclust`.

Usage

```
## S3 method for class 'lsbclust'
plot(x, type = c("overall", "rows", "columns",
  "interactions"), biplot.axes = TRUE, ...)
```

Arguments

x	An object of class orc.kmeans
type	A character vector indicating which component(s) of x to plot: a combination of "overall", "rows", "columns" and "interactions".
biplot.axes	A logical indicating whether to plot calibrated biplot axes for the line segments indicated in segments or not.
...	additional arguments passed to the plot methods of the respective components, typically to theme. Use e.g. plot(x$interactions) for more control over the respective plots.

Author(s)

Pieter C. Schoonees

See Also

plot.int.lsbclust, plot.ovl.kmeans, plot.row.kmeans, plot.col.kmeans

Examples

```
data("dcars")
m <- lsbclust(data = dcars, margin = 3, delta = c(1, 1, 1, 1), nclust = 5, nstart = 1)
plot(m)
```

---

plot.ovl.kmeans *Plot method for class 'ovl.kmeans'*

Description

Simple plot method for object of class 'ovl.kmeans' as output by orc.lsbclust.

Usage

```
## S3 method for class 'ovl.kmeans'
plot(x, which = 1L, ...)
```

Arguments

x	An object of class ovl.kmeans
which	Which type of plot to produce. Currently only which = 1 is implemented.
...	additional arguments passed to theme.

Author(s)

Pieter C. Schoonees

Examples

```
data("dcars")
m <- orc.lsbclust(data = dcars, margin = 3, delta = c(1,1,1,1), nclust = 5, type = "overall")
plot(m)
```

---

plot.row.kmeans *Plot method for class 'row.kmeans'*

Description

Simple plot method for object of class 'row.kmeans' as output by orc.lsbclust.

Usage

```
## S3 method for class 'row.kmeans'
plot(x, which = 1L, ...)
```

Arguments

x	An object of class row.kmeans
which	Which type of plot to produce (only 3 types are implemented).
...	additional arguments passed to theme.

Author(s)

Pieter C. Schoonees

Examples

```
data("dcars")
m <- orc.lsbclust(data = dcars, margin = 3, delta = c(1,1,1,1), nclust = 5, type = "rows")
plot(m)
```

---

plot.step.lsbclust *Plot method for class 'step.lsbclust'*

Description

Plot 'step.lsbclust' objects.

Usage

```
## S3 method for class 'step.lsbclust'
plot(x, which = 1L:5L, col.all = NULL,
  arrange = FALSE, chull = FALSE, ...)
```

Arguments

x	An object of class step.lsbclust
which	Which type of plot to produce.
col.all	A character vector of length one indicating which of "overall", "rows", "columns" or "interactions" should be mapped to colour in the plot for all possible models. Care needs to be taken that the stated component is included in the fit.
arrange	Logical indicating whether the arrange the plots side-by-side via grid.arrange or not.
chull	Logical indicating whether to plot the estimated convex hull or not.
...	additional arguments passed to theme.

Author(s)

Pieter C. Schoonees

---

print.lsbclust *Print method for object of class 'lsbclust'*

Description

Print a 'lsbclust' object.

Usage

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

Arguments

x	An object of class 'lsbclust'
...	Unimplemented.

---

sim.lsbclust *Simulate from an LSBLCUST model*

Description

This is a simple functiomn to simulate data from an LSBCLUST model.

Usage

```
sim.lsbclust(N = 100, nclust = 5, J = 10, K = 8, ndim = 2, sd = 1)
```

Arguments

N	Integer giving the number of observations required.
nclust	Integer giving the number of clusters required.
J	Integer giving the number of rows to sample.
K	Integer giving the number of columns ot sample.
ndim	Integer giving the desired dimensionality of the sampled cluster means.
sd	The sd argument of rnorm.

---

step.lsbclust *Model Search for lsbclust*

Description

Fit lsbclust models for different numbers of clusters and/or different values of delta. The resulting output can be inspected through its plot method to facilitate model selection. Each component of the model is fitted separately.

Usage

```
step.lsbclust(data, margin = 3L, delta = c(1, 1, 1, 1), nclust, ndim = 2,
  fixed = c("none", "rows", "columns"), nstart = 20, starts = NULL,
  nstart.kmeans = 500, alpha = 0.5, parallelize = FALSE, maxit = 100,
  verbose = -1, type = NULL, ...)
```

Arguments

data	A three-way array representing the data.
margin	An integer giving the single subscript of data over which the clustering will be applied.
delta	A four-element binary vector (logical or numeric) indicating which sum-to-zero constraints must be enforced.
nclust	Either a vector giving the number of clusters which will be applied to each element of the model, that is to (a subset of) the overall mean, row margins, column margins and interactions. If it is a list, arguments are matched by the names "overall", "rows" "columns" and "interactions". If the list does not have names, the components are extracted in the aforementioned order.
ndim	The required rank for the approximation of the interactions (a scalar).
fixed	One of "none", "rows" or "columns" indicating whether to fix neither sets of coordinates, or whether to fix the row or column coordinates across clusters respectively. If a vector is supplied, only the first element will be used (passed to int.lsbclust).
nstart	The number of random starts to use for the interaction clustering.

starts	A list containing starting configurations for the cluster membership vector. If not supplied, random initializations will be generated (passed to int.lsbclust).
nstart.kmeans	The number of random starts to use in kmeans.
alpha	Numeric value in [0, 1] which determines how the singular values are distributed between rows and columns (passed to int.lsbclust).
parallelize	Logical indicating whether to parallelize over different starts or not (passed to int.lsbclust).
maxit	The maximum number of iterations allowed in the interaction clustering.
verbose	The number of iterations after which information on progress is provided (passed to int.lsbclust).
type	One of "rows", "columns" or "overall" (or a unique abbreviation of one of these) indicating whether clustering should be done on row margins, column margins or the overall means of the two-way slices respectively. If more than one opion are supplied, the algorithm is run for all (unique) options supplied (passed to orc.lsbclust). This is an optional argument.
...	Additional arguments passed to kmeans.

Examples

```
m <- step.lsbclust(data = dcars, margin = 3, delta = c(1, 0, 1, 0), nclust = 4:5,
                    ndim = 2, fixed = "columns", nstart = 1, nstart.kmeans = 100,
                    parallelize = FALSE)

## For a list of all deltas
delta <- expand.grid(replicate(4, c(0,1), simplify = FALSE))
delta <- with(delta, delta[!(Var1 == 0 & Var3 == 1), ])
delta <- with(delta, delta[!(Var2 == 0 & Var4 == 1),])
delta <- delta[-4,]
delta <- as.list(as.data.frame(t(delta)))
m2 <- step.lsbclust(data = dcars, margin = 3, delta = delta, nclust = 4:5,
                    ndim = 2, fixed = "columns", nstart = 1, nstart.kmeans = 100,
                    parallelize = FALSE)
```

---

summary.int.lsbclust *Summary Method for Class "int.lsbclust"*

Description

Some goodness-of-fit diagnostics are provided for all three margins.

Usage

```
## S3 method for class 'int.lsbclust'
summary(object, digits = 3, ...)
```

Arguments

object	An object of class 'int.lsbclust'.
digits	The number of digits in the printed output.
...	Unimplemented.

---

summary.lsbclust *Summary Method for Class "lsbclust"*

Description

Summarize a lsbclust object.

Usage

```
## S3 method for class 'lsbclust'
summary(object, digits = 3, ...)
```

Arguments

object	An object of class 'lsbclust'.
digits	The number of digits in the printed output.
...	Unimplemented.

---

supermarkets *Dutch Supermarkets Data Set*

Description

This data set relates to 220 consumers rating 10 Dutch supermarket chains according to 8 variables. A rating scale from 1 to 10 was used.

Usage

```
supermarkets
```

Format

A three-way array with supermarkets in the first dimension, variables in the second and consumers in the third dimension.

Source

Michel van de Velden

Examples

```
data("supermarkets")
fit <- lsbclust(data = supermarkets, nclust = 6, fixed = "rows", nstart = 2)
```

T3Clusf *T3Clusf: Tucker3 Fuzzy Cluster Analysis*

Description

This is an implementation of the T3Clusf algorithm of Rocci & Vichi (2005).

Usage

```
T3Clusf(X, Q, R = Q, G = 2, margin = 3L, alpha = 1, eps = 1e-08,
  maxit = 100L, verbose = 1, nr.starts = 1L, parallel = TRUE,
  mc.cores = detectCores() - 1)
```

Arguments

X	Three-way data array, with no missing values.
Q	Integer giving the number of dimensions required for mode B (variables). This is the first mode of the array, excluding the mode clustered over (see margin).
R	Integer giving the number of dimensions required for mode C (occasions). This is the second mode of the array, excluding the mode clustered over (see margin).
G	Integer giving the number of clusters required.
margin	Integer giving the margin of the array to cluster over. The remaining two modes, in the original order, corresponds to Q and R.
alpha	Numeric value giving the fuzziness parameter.
eps	Small numeric value giving the empirical convergence threshold.
maxit	Integer giving the maximum number of iterations allowed.
verbose	Integer giving the number of iterations after which the loss valus is printed.
nr.starts	Integer giving the number of random starts required.
parallel	Logical indicating whether to parallelize over random starts if nr.starts > 1.
mc.cores	Argument passed to mclapply.

References

Rocci, R., & Vichi, M. (2005). *Three-mode component analysis with crisp or fuzzy partition of units*. Psychometrika, 70(4), 715-736.

Examples

```
data("dcars")
set.seed(13)
res <- T3Clusf(X = dcars, Q = 3, R = 2, G = 3, alpha = 2)
```

Index