Package ‘lga’

February 20, 2015

Version 1.1-1
Date 2008-06-15
Title Tools for linear grouping analysis (LGA)
Author Justin Harrington
Maintainer ORPHANED
Depends R (>= 2.2.1)
Imports boot, lattice
Suggests snow, MASS, maps
Description Tools for linear grouping analysis. Three user-level functions: gap, rlga and lga.
License GPL
X-CRAN-Comment Orphaned on 2012-01-10 as maintainer address bounces and maintainer is untraceable.
Repository CRAN
Date/Publication 2012-01-13 08:52:30
NeedsCompilation no

R topics documented:

  lga-package .................................................. 2
  brain ....................................................... 2
  corridorWalls ............................................. 3
  gap ........................................................ 3
  lga ........................................................ 6
  nhl94 ...................................................... 9
  ob ......................................................... 9

Index 11
The lga Package

Description

The lga package is an implementation of the algorithms described in Van Aelst et al (2006) and Garcia-Escudero et al (2008). It has three main functions (with accompanying print and plot methods).

- **lga** The core linear grouping analysis
- **rlga** The robust version of linear grouping analysis
- **gap** Performs a gap analysis to find the number of clusters

For more details refer to the relevant help files.

Author(s)

Justin Harrington <harringt@stat.ubc.ca>

References


Usage

data(brain)

Format

A matrix with 282 observations on the following 2 variables:

- **BrainWeight**: g.
- **OlfactoryBulbs**: ml.
corridorWalls

References

corridorWalls Corridor Walls data

Description
Corridor Walls data from Garcia-Escudero et al. (2008). A laser device is introduced in a corridor of an office building and throws a laser ray which touches a point from the object found at the end of its trajectory. The device produces a three dimensional measurement of the placement of that point with respect to a fixed reference system.

Usage
data(corridorWalls)

Format
A matrix with 11710 observations on the following 3 variables (3-dimensional coordinates).

x a numeric vector
y a numeric vector
z a numeric vector

References

gap Perform gap analysis

Description
Performs the gap analysis using lga to estimate the number of clusters.

Usage
## Default S3 method:
gap(x, K, B, criteria=c("tibshirani", "DandF","none"),
  nnode=NULL, scale=TRUE, ...)

Arguments

- **x**: a numeric matrix.
- **k**: an integer giving the maximum number of clusters to consider.
- **B**: an integer giving the number of bootstraps.
- **criteria**: a character string indicating which criteria to evaluate the gap data. One of “'tibshirani'” (default), “'DandF'” or “'none'”. Can be abbreviated.
- **nnode**: an integer of many CPUS to use for parallel processing. Defaults to NULL i.e. no parallel processing.
- **scale**: logical. Should the data be scaled?
- **...**: For any other arguments passed from the generic function.

Details

This code performs the gap analysis using lga. The gap statistic is defined as the difference between the log of the Residual Orthogonal Sum of Squared Distances (denoted \( \log(W_k) \)) and its expected value derived using bootstrapping under the null hypothesis that there is only one cluster. In this implementation, the reference distribution used for the bootstrapping is a random uniform hypercube, transformed by the principal components of the underlying data set. For further details see Tibshirani et al (2001).

For different criteria, different rules apply. With “'tibshirani'” (ibid) we calculate the gap statistic for \( k = 1, \ldots, K \), stopping when

\[
gap(k) \geq \gap(k + 1) - s_{k+1}
\]

where \( s_{k+1} \) is a function of standard deviation of the bootstrapped estimates.

With the “'DandF'” criteria from Dudoit et al (2002), we calculate the gap statistic for all values of \( k = 1, \ldots, K \), selecting the number of clusters as

\[
\hat{k} = \text{smallest } k \geq 1 \text{ such that } \gap(k) \geq \gap(k^*) - s_{k^*}
\]

where \( k^* = \arg \max_{k \geq 1} \gap(k) \).

Finally, for the criteria “none”, no rules are applied, and just the gap data is returned.

As lga is ostensibly unsupervised in this case, the parameter niter is set to 20 to ensure convergence.

This function is parallel computing aware via the nnode argument, and works with the package snow. In order to use parallel computing, one of MPI (e.g. lamboot) or PVM is necessary. For further details, see the documentation for snow.

Value

An object of class “'gap'” with components

- **finished**: a logical. For the “tibshirani”, was there a solution found?
- **nclust**: a integer for the number of clusters estimated. Returns NA if nothing conclusive is found.
- **data**: the original data set, scaled if specified in the arguments.
- **criteria**: the criteria used.
Author(s)
Justin Harrington <harringt@stat.ubc.ca>

References

See Also
lga

Examples

```r
## Synthetic example
## Make a dataset with 2 clusters in 2 dimensions

library(MASS)
set.seed(1234)
X <- rbind(mvrnorm(n=100, mu=c(1, -2), Sigma=diag(0.1, 2) + 0.9),
           mvrnorm(n=100, mu=c(1, 1), Sigma=diag(0.1, 2) + 0.9))
gap(X, k=4, B=20)

## to run this using parallel processing with 4 nodes, the equivalent
## code would be

## Not run: gap(X, K=4, B=20, nnode=4)

## Quakes data (from package:datasets)
## Including the first two dimensions versus three dimensions
## yields different results

set.seed(1234)
## Not run:
gap(quakes[,1:2], K=4, B=20)
gap(quakes[,1:3], K=4, B=20)

## End(Not run)

library(maps)
lgaout1 <- lga(quakes[,1:2], k=3)
plot(lgaout1)

lgaout2 <- lga(quakes[,1:3], k=2)
```
lga

Description

Linear Grouping Analysis

Usage

## Default S3 method:

lga(x, k, biter = NULL, niter = 10, showall = FALSE, scale = TRUE, 
nnode=NULL, silent=FALSE, ...)

## Default S3 method:

rlga(x, k, alpha=0.9, biter = NULL, niter = 10, showall = FALSE, scale = TRUE, 
nnode=NULL, silent=FALSE, ...)

Arguments

x 
a numeric matrix.

k 
an integer for the number of clusters.

alpha 
a numeric value between 0.5 and 1. For the robust estimate of LGA, specifying 
the percentage of points in the best subset.

biter 
an integer for the number of different starting hyperplanes to try.

niter 
an integer for the number of iterations to attempt for convergence.

showall 
logical. If TRUE then display all the outcomes, not just the best one.

scale 
logical. Allows you to scale the data, dividing each column by its standard 
deviation, before fitting.

nnode 
an integer of many CPUS to use for parallel processing. Defaults to NULL i.e. 
no parallel processing.

silent 
logical. If TRUE, produces no text output during processing.

... 
For any other arguments passed from the generic function.
Details

This code tries to find k clusters using the lga algorithm described in Van Aelst et al (2006). For each attempt, it has up to niter steps to get to convergence, and it does this from biter different starting hyperplanes. It then selects the clustering with the smallest Residual Orthogonal Sum of Squareds.

If biter is left as NULL, then it is selected via the equation given in Van Aeslt et al (2006).

The function rlga is the robust equivalent to LGA, and is introduced in Garcia-Escudero et al (2008).

Both functions are parallel computing aware via the nnode argument, and works with the package snow. In order to use parallel computing, one of MPI (e.g. lamboot) or PVM is necessary. For further details, see the documentation for snow.

Associated with the lga and rlga functions are a print method and a plot method (see the examples). In the plot method, the fitted hyperplanes are also shown as dashed-lines when there are only two dimensions.

Value

An object of class “lga”. The list contains

cluster a vector containing the cluster memberships.
ROSS the Residual Orthogonal Sum of Squares for the solution.
converged a logical. True if at least one solution has converged.
nconverg the number of converged solutions (out of biter starts).
x the (scaled if selected) dataset.

and the attributes include

scaled logical. Is the data scaled?
k the number of clusters to be found.
biter the biter setting used.
niter the niter setting used.

Author(s)

Justin Harrington <harringt@stat.ubc.ca>

References


See Also

gap
Examples

## Synthetic Data
## Make a dataset with 2 clusters in 2 dimensions

```r
library(MASS)
set.seed(1234)
X <- rbind(mvrnorm(n=100, mu=c(1,-1), Sigma=diag(0.1,2)+0.9),
           mvrnorm(n=100, mu=c(1,1), Sigma=diag(0.1,2)+0.9))

lgaout <- lga(X,2)
plot(lgaout)
print(lgaout)
```

## Robust equivalent

```r
rlgaout <- rlga(X,2, alpha=0.75)
plot(rlgaout)
print(rlgaout)
```

## nhl94 data set

```r
data(nhl94)
plot(lga(nhl94, k=3, niter=30))
```

## Allometry data set

```r
data(brain)
plot(lga(log(brain, base=10), k=3))
```

## Second Allometry data set

```r
data(ob)
plot(lga(log(ob[,2:3]), k=3), pch=as.character(ob[,1]))
```

## Corridor Walls data set
## To obtain the results reported in Garcia-Escudero et al. (2008):

```r
data(corridorWalls)
rlgaout <- rlga(corridorWalls, k=3, biter = 100, niter = 30, alpha=0.85)
pairs(corridorWalls, col=rlgaout$cluster+1)
plot(rlgaout)
```

## Parallel processing case
## In this example, running using 4 nodes.

## Not run:

```r
set.seed(1234)
X <- rbind(mvrnorm(n=1e6, mu=c(1,-1), Sigma=diag(0.1,2)+0.9),
           mvrnorm(n=1e6, mu=c(1,1), Sigma=diag(0.1,2)+0.9))
abc <- lga(X, k=2, nnode=4)
```
### nhl94

*Player Performance in NHL for 1994-1995*

#### Description

This data set gives four variables for each of 871 players for the NHL 1994-1995 season. From *Van Aelst et al* (2006)

#### Usage

```r
data(nhl94)
```

#### Format

A matrix with 871 observations on the following 3 variables:

- **PTS**: Points scored, a numeric vector
- **PM**: Plus/minus average rating, a numeric vector
- **PIM**: Total penalty time in minutes, a numeric vector
- **PP**: Power play goals, a numeric vector

#### References


### ob

*Allometry Data*

#### Description

Allometry data from *Van Aelst et al* (2006)

#### Usage

```r
data(ob)
```

#### Format

A data frame with 83 observations on the following 3 variables.

- **Group**: a factor with levels `a`, `c`, `h`, `i`, `m`, `p`, `t`
- **BrainWeight**: a numeric vector
- **OlfactoryBulbs**: a numeric vector
References

Index

*Topic **cluster**
  gap, 3
  lga, 6
  lga-package, 2

*Topic **datasets**
  brain, 2
  corridorWalls, 3
  nhl94, 9
  ob, 9

*Topic **multivariate**
  gap, 3
  lga, 6
  lga-package, 2

brain, 2

corridorWalls, 3

gap, 2, 3, 7

lga, 2, 5, 6
  lga-package, 2

nhl94, 9

ob, 9

rlga, 2
  rlga(lga), 6