Package ‘RPS’

July 27, 2018

Type Package
Title Resistant Procrustes Superimposition
Version 1.0.1
Maintainer Guillermo Andres Pacheco <guillermopacheco.exa@gmail.com>
Description Based on RPS tools, a rather complete resistant shape analysis of 2D and 3D datasets based on landmarks can be performed. In addition, landmark-based resistant shape analysis of individual asymmetry in 2D for matching or object symmetric structures is also possible.
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 6.0.1
Imports geomorph, MASS, igraph, ape, matlab, Gmedian
NeedsCompilation no
Author Guillermo Andres Pacheco [aut, cre],
    Sebastian Torcida [aut],
    Viviana Ferraggine [aut],
    Federico Lotto [aut]
Repository CRAN
Date/Publication 2018-07-27 21:30:05 UTC

R topics documented:

‘RPS’ ................................................................. 2
cmdistance_RPS .................................................. 3
eucunivMDS_RPS ................................................ 4
matchingsymm_RPS ............................................. 5
objectsymm_RPS ................................................. 5
procrustesCM_RPS ............................................. 6
readlandtxtMorphJ_RPS ...................................... 7
resdistance_RPS ................................................ 8
resunivMDS_RPS ................................................ 9
Description

RPS provides a set of tools to perform a rather complete descriptive landmark-based resistant shape analysis 3D and 2D, following Torcida et al. 2014 ("An integrated approach for landmark-based resistant shape analysis in 3D", Evol. Biol. 41(2):351_366). More specifically, these tools enable to obtain: i) a generalized resistant Procrustes superposition (robgit_RPS.R) for a set of configurations of landmarks either in 3D and 2D; ii) a resistant distance (resdistance_RPS.R) to quantify shape differences obtained following the resistant Procrustes superimposition, and iii) a resistant ordination (resunivMDS_RPS.R) of the superimposed configurations based on the universal Multidimensional Scaling from (Agarwal et al. 2010). Corresponding least squares (LS) counterparts of all these tools (procrustesCM_RPS.R, cmdistance_RPS.R and eucunivMDS_RPS.R, respectively) have also been implemented in RPS_R to offer a more complete and self-contained set of shape analysis descriptive tools. This enables the comparison of the LS and resistant superimposition results when applied to the same dataset. Also included is a rather new method for a resistant analysis of individual shape asymmetry for configurations of landmarks in 2D with bilateral symmetry (matching or object symmetry), following Torcida et al. 2016 ("A resistant method for landmark-based analysis of individual asymmetry in two dimensions", Quant. Biol. 4(4):270_282). The main tools enable to estimate the resistant symmetric shape under matching symmetry (matchingsymm_RPS.R) and the resistant symmetric shape estimation under object symmetry (objectsymm_RPS.R). In both cases, a plot of the results and the table sof landmarks contributions to asymmetry are also offered.

Usage

robgit_RPS(X, consenso = FALSE)

Arguments

X       A s-dimensional array of n x k matrices (k configurations of n landmarks), each representing the shape of an object
consenso A logical value that determines if the consensus configuration is returned.

Value

s-dimensional array of n x k matrices, representing the (resistant) superimposed objects

Functions

eucunivMDS_RPS, resunivMDS_RPS, cmdistance_RPS, resdistance_RPS, readlandtxtMorphJ_RPS, robgit_RPS, matchingsymm_RPS, objectsymm_RPS, procrustesCM_RPS

Author(s)

Guillermo Pacheco, Viviana Ferraggine, Sebastian Torcida
cmdistance_RPS

Examples

```r
source = array(matrix(nrow = 8, ncol = 3), c(8, 3, 3), dimnames = NULL)
source[, 1] <- matrix(c(3, 0, 0, 3, 0, 1, 3, 1, 1, 3, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0),
                    nrow = 8, ncol = 3, byrow = TRUE)
source[, 2] <- matrix(c(3, 0, 3, 0, 0.5, 3, 1, 0.75, 3, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0.25),
                    nrow = 8, ncol = 3, byrow = TRUE)
source[, 3] <- matrix(c(5, 2, 1, 3, 0, 1.5, 3, 4, 1, 1.75, 3, 1, 0, 0, 0, 0, 2, 1, 0, 3, 1, 0, 1, 0.75),
                    nrow = 8, ncol = 3, byrow = TRUE)
result <- RPS::robgit_RPS(source, consenso = FALSE)
RPS::cmdistance_RPS(result)
```

Description

This function computes the least-squares Procrustes distance between each pair of matrices (configurations of landmarks) from the input set

Usage

```r
cmdistance_RPS(X)
```

Arguments

- `X` The input set of nx3 matrices (objects)

Value

The LS Procrustes distance matrix between pairs of objects

Author(s)

Guillermo Pacheco, Viviana Ferraggine, Sebastian Torcida

Examples

```r
source = array(matrix(nrow = 8, ncol = 3), c(8, 3, 3), dimnames = NULL)
source[, 1] <- matrix(c(3, 0, 0, 3, 0, 1, 3, 1, 1, 3, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0),
                    nrow = 8, ncol = 3, byrow = TRUE)
source[, 2] <- matrix(c(3, 0, 3, 0, 0.5, 3, 1, 0.75, 3, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0.25),
                    nrow = 8, ncol = 3, byrow = TRUE)
source[, 3] <- matrix(c(5, 2, 1, 3, 0, 1.5, 3, 4, 1, 1.75, 3, 1, 0, 0, 0, 0, 2, 1, 0, 3, 1, 0, 1, 0.75),
                    nrow = 8, ncol = 3, byrow = TRUE)
result <- RPS::robgit_RPS(source)
RPS::cmdistance_RPS(result)
```
Given a $n \times n$ distance matrix $D$ (not necessarily Euclidean) and an initial set $X_0$ of $n$ seeds in $k$ dim (that is, an initial $n \times k$ matrix), this function finds a set of $n$ points in $k$ dimensions $X$ (a final $n \times k$ matrix) through a least-squares criterion such that the $n \times n$ matrix $D_k$ of euclidean distances among these new points $X$ is as close as possible to $D$.

**Usage**

eucunivMDs_RPS(D, k = 2)

**Arguments**

- **D** distance matrix $n \times n$ to be approximated
- **k** dimension of output results

**Value**

- **X** A set of $n$ points in $k$ dimensions

**Author(s)**

Guillermo Pacheco, Viviana Ferraggine, Sebastian Torcida

**Examples**

```r
source = array(matrix(nrow = 8, ncol = 3), c(8, 3, 3), dimnames = NULL)
source[,1] <- matrix(c(3, 0, 0, 3, 0, 1, 3, 1, 1, 3, 1, 3, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0),
                     nrow = 8, ncol = 3, byrow = TRUE)
source[,2] <- matrix(c(3, 0, 0, 3, 0, 0, 5, 5, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0.25),
                     nrow = 8, ncol = 3, byrow = TRUE)
source[,3] <- matrix(c(5, 2, 1, 3, 0, 1.5, 3.4, 1, 1.75, 3, 1, 0, 0, 0, 0, 0, 2, 1, 0, 3, 1, 0, 1, 0.75),
                     nrow = 8, ncol = 3, byrow = TRUE)
result <- RPS::robgit_RPS(source, consenso = FALSE)
distance <- RPS::resdistance_RPS(result)
RPS::eucunivMDs_RPS(distance, 2)
```
matchingsymm_RPS

This function obtains the individual resistant-symmetric shape for 2D matching-symmetry data. The input is an array A of size n (landmarks) x p (dimensions) x 2k (objects: the left-right sides for each). Configurations are ordered in this way: left side Object 1, right side Object 1, left side Object 2, right side Object 2, etc.

Usage

matchingsymm_RPS(A, ctr="gmedian", legend.loc="topleft")

Arguments

A
an array of size n (landmarks) x 2 (in 2D) x 2k (left/right sides for k configurations)

ctr
Centering options: "gmedian" (the spatial or geometric median, default choice), "median" (the componentwise median), "mean" (the average)

legend.loc
The location of the legend for the plot

Author(s)

Federico Lotto, Sebastian Torcida

objectsymm_RPS

This function obtains the individual resistant-symmetric shape for 2D object-symmetry data. The input is an array A of size n (landmarks) x p (dimensions) x k (objects) Landmarks must be in this order: saggital (or unpaired) landmarks first, then left paired landmarks and finally right paired landmarks. Configurations are ordered in this way: L side Object 1 and R side Object 1, L side Object 2 and R side Object 2, etc.

Description

This function obtains the individual resistant-symmetric shape for 2D object-symmetry data. The input is an array A of size n (landmarks) x p (dimensions) x k (objects) Landmarks must be in this order: saggital (or unpaired) landmarks first, then left paired landmarks and finally right paired landmarks. Configurations are ordered in this way: L side Object 1 and R side Object 1, L side Object 2 and R side Object 2, etc.
Usage

```r
objectsymm_RPS(A, ctr="gmedian", prs.file, proj.met="msum", legend.loc="topleft")
```

Arguments

- **A**: Input data: an array or matrix of size n (landmarks) x 2 (in 2D) x k (objects)
- **ctr**: Centering options: "gmedian" (the spatial or geometric median, default choice), "median" (the componentwise median), "mean" (the average)
- **prs.file**: This is a .txt file indicating the L+R paired landmarks as rows: e.g. 7 15; 8 16; etc.
- **proj.met**: The choice to compute the saggital axis: sum or median of projections
- **legend.loc**: The location of the legend for the plot.result function

Value

w

Author(s)

Federico Lotto, Sebastian Torcida

---

**procrustesCM_RPS**

This function is simply a wrapper for the geomorph function gpagen that performs the classical least squares Procrustes superimposition of the input configurations of landmarks.

Description

This function is simply a wrapper for the geomorph function gpagen that performs the classical least squares Procrustes superimposition of the input configurations of landmarks.

Usage

```r
procrustesCM_RPS(X)
```

Arguments

- **X**: A s-dimensional array (s=2 or s=3) of n x k matrices, representing shapes of k objects through n landmarks in s dimensions

Value

s-dimensional array of n x k matrices, representing shapes of k objects following superimposition.

Author(s)

Dean C. Adams, Michael Collyer
Examples

source = array(matrix(nrow = 8,ncol = 3),c(8,3,3),dimnames = NULL)
source[,1] <- matrix(c(3,0,3,0,1,3,1,1,1,3,1,0,0,0,0,1,0,1,0,1,0),
   ,nrow = 8,ncol = 3,byrow = TRUE)
source[,2] <- matrix(c(3,0,0,3,0,0,5,3,1,0.75,3,1,0,0,0,0,1,0,1,0,1,0.25),
   ,nrow = 8,ncol = 3,byrow = TRUE)
source[,3] <- matrix(c(5,2,1,3,0,1.5,3,1,1.75,3,1,0,0,0,0,0,2,1,0,3,1,0,1,0.75),
   ,nrow = 8,ncol = 3,byrow = TRUE)
result <- RPS::procrustesCM_RPS(source)
result

Description

Reads a MorphoJ .txt file and returns it as an array of n x k matrices in s dimensions (s=2 or s=3)

Usage

readlandtxtMorphJ_RPS(path, dim)

Arguments

path  Path of file
dim   Dimension of the data (2D or 3D).

Value

A s-dimensional array of n x k matrices and a list of the corresponding object’s names

Author(s)

Guillermo Pacheco, Viviana Ferraggine, Sebastian Torcida
This function computes the resistant distance between each pair of matrices from the input set.

**Description**

This function computes the resistant distance between each pair of matrices from the input set.

**Usage**

```
resdistance_RPS(x)
```

**Arguments**

- **x**
  
The input set of nx3 matrices (objects)

**Value**

This function computes the sum of non-squared euclidean distances across landmarks for each pair of matrices from the input set.

**Author(s)**

Guillermo A. Pacheco, Viviana Ferragignie, Sebastian Torcida

**Examples**

```r
code
```
resunivMDS_RPS

Given a $n \times n$ distance matrix $D$ (not necessarily Euclidean) and an initial set $X_0$ (that is, a $n \times k$ matrix) of $n$ seeds in $k$ dim, this function finds a set of $n$ points in $k$ dimensions $X$ (that is, a $k \times n$ matrix) using a resistant criterion such that the $n \times n$ matrix $D_k$ of euclidean distances among these new points $X$ is as close as possible to $D$.

Description

Given a $n \times n$ distance matrix $D$ (not necessarily Euclidean) and an initial set $X_0$ (that is, a $n \times k$ matrix) of $n$ seeds in $k$ dim, this function finds a set of $n$ points in $k$ dimensions $X$ (that is, a $k \times n$ matrix) using a resistant criterion such that the $n \times n$ matrix $D_k$ of euclidean distances among these new points $X$ is as close as possible to $D$.

Usage

```r
resunivMDS_RPS(D,k)
```

Arguments

- **D**: distance matrix $n \times n$ to be approximated
- **k**: dimension of output results

Value

- **X**: A set of $n$ points in $k$ dimensions

Author(s)

Guillermo Pacheco, Viviana Ferraggine, Sebastian Torcida

Examples

```r
source = array(matrix(nrow = 8,ncol = 3),c(8,3,3),dimnames = NULL)
source[,1] <- matrix(c(3,0,0,3,0,1,3,1,1,3,1,1,0,0,0,0,0,0,1,0,1,0,1,0),
                    nrow = 8,ncol = 3,byrow = TRUE)
source[,2] <- matrix(c(3,0,0,3,0,0.5,3,1,0,0,0,0,0,1,0,1,0,1,0,1,0,25),
                    nrow = 8,ncol = 3,byrow = TRUE)
source[,3] <- matrix(c(5,2,1,3,0,1.5,3,4,1,1.75,3,1,0,0,0,0,0,2,1,0,3,1,0,1,0.75),
                    nrow = 8,ncol = 3,byrow = TRUE)
result <- RPS::robgit_RPS(source, consenso = FALSE)
distance <- RPS::resdistance_RPS(result)
RPS::resunivMDS_RPS(distance,2)
```
Index

'RPS', 2
'RPS'-package ('RPS'), 2

cmdistance_RPS, 3

eucunivMDS_RPS, 4

matchingsymm_RPS, 5

objectsymm_RPS, 5

procrustesCM_RPS, 6

readlandtxtMorphJ_RPS, 7

resdistance_RPS, 8

resunivMDS_RPS, 9

robgit_RPS ('RPS'), 2