Package ‘geomorph’

October 27, 2015

Date 2015-10-26
Type Package
Title Geometric Morphometric Analyses of 2D/3D Landmark Data
Version 2.1.7-1
Author Dean Adams, Michael Collyer, Emma Sherratt
Maintainer Dean Adams <dCADAMs@iastate.edu>
Depends rgl, ape, R(>= 3.1.0)
Imports graphics, grDevices, stats, utils, jpeg, geiger, phytools, Matrix
Description Read, manipulate, and digitize landmark data, generate shape variables via Procrustes analysis for points, curves and surfaces, perform shape analyses, and provide graphical depictions of shapes and patterns of shape variation.
License GPL (>= 2)
NeedsCompilation yes
Repository CRAN
Date/Publication 2015-10-27 09:52:43

R topics documented:

geomorph-package .................................................. 3
advanced.procD.lm .................................................. 3
arrayspecs .............................................................. 6
bilat.symmetry ......................................................... 7
buildtemplate .......................................................... 9
compare.evol.rates ................................................... 11
compare.modular.partitions .......................................... 13
compare.multi.evol.rates ............................................. 14
define.links .......................................................... 16
define.modules ....................................................... 17
define.sliders ......................................................... 18
digit.curves ............................................................ 20
**Description**

Geometric morphometric analyses for 2D/3D data

**Author(s)**

Dean C. Adams, Michael Collyer, & Emma Sherratt

Functions in this package allow one to read, manipulate, and digitize landmark data; generate shape variables via Procrustes analysis for points, curves and surface data, perform statistical analyses of shape variation and covariation, and provide graphical depictions of shapes and patterns of shape variation.

---

**advanced.procD.lm**

Procrustes ANOVA and pairwise tests for shape data, using complex linear models

**Description**

The function quantifies the relative amount of shape variation explained by a suite of factors and covariates in a "full" model, after accounting for variation in a "reduced" model. Inputs are formulae for full and reduced models (order is not important), plus indication if means or slopes are to be compared among groups, with appropriate formulae to define how they should be compared.

**Usage**

```r
advanced.procD.lm(f1, f2, groups = NULL, slope = NULL, angle.type = c("r", "deg", "rad"), iter = 999, verbose = FALSE, ...)
```

**Arguments**

- `f1`: A formula for a linear model, containing the response matrix (e.g., `y ~ x1 + x2`)
- `f2`: A formula for another linear model (e.g., `~ x1 + x2 + x3 + a*b`) (f1 and f2 should be nested)
- `groups`: A formula for grouping factors (e.g., `~ a`, or `~ a*b`)
- `slope`: A formula with one covariate (e.g., `~ x3`)
- `angle.type`: A value specifying whether differences between slopes should be represented by vector correlations (`r`), radians (rad) or degrees (deg)
- `iter`: Number of iterations for significance testing
- `verbose`: A logical value specifying whether additional output should be displayed (see Value below)
- `...`: Arguments passed on to procD.fit (typically associated with the lm function)
Details

The response matrix 'y' can be in the form of a two-dimensional data matrix of dimension (n x (p x k)) or a 3D array (p x k x n). It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with \texttt{gpagen}]. The names specified for the independent (x) variables in the formula represent one or more vectors containing continuous data or factors. It is assumed that the order of the specimens in the shape matrix matches the order of values in the independent variables. Linear model fits (using the \texttt{lm} function) can also be input in place of a formula. Arguments for \texttt{lm} can also be passed on via this function.

The function performs statistical assessment of the terms in the model using Procrustes distances among specimens, rather than explained covariance matrices among variables. With this approach, the sum-of-squared Procrustes distances are used as a measure of SS (see Goodall 1991). The SS between models is evaluated through permutation. In morphometrics this approach is known as a Procrustes ANOVA (Goodall 1991), which is equivalent to distance-based anova designs (Anderson 2001). Unlike \texttt{procD.lm}, this function is strictly for comparison of two nested models. The function will readily accept non-nested models, but the results will not be meaningful. (Use of \texttt{procD.lm} will be more suitable in most cases.) A residual randomization permutation procedure (RRPP) is utilized for reduced model residuals to evaluate the SS between models (Collyer et al. 2015). Effect-sizes (Z-scores) are computed as standard deviates of the SS sampling distributions generated, which might be more intuitive for P-values than F-values (see Collyer et al. 2015).

Pairwise tests are only performed if formulae are provided to compute such results.

Value

Function returns an ANOVA table of statistical results for model comparison: error df (for each model), SS, MS, F ratio, Z, and Prand. The following may also be returned.

- \texttt{Means.dist} Pairwise distance between means, if applicable
- \texttt{LS.Mean.dist} Pairwise distance between LS means, if applicable
- \texttt{Prob.Mean.dist} P-values for pairwise distances between means
- \texttt{Slopes.dist} Pairwise distance between slope vectors (difference in amount of shape change), if applicable
- \texttt{Prob.Slopes.dist} P-values for pairwise distances between slopes
- \texttt{Slopes.correlation} Pairwise vector correlations between slope vectors, if applicable
- \texttt{Prob.Slopes.cor} P-values for pairwise correlations between slope vectors (high correlation less significant)
- \texttt{Slopes.angle} Angles between slope vectors, if applicable
- \texttt{Prob.Slopes.angle} P-values for pairwise angles between slope vectors
- \texttt{SS.rand} Random SS from RRPP permutations (when \texttt{verbose=TRUE})
- \texttt{random.mean.dist} Random pairwise distances between means from RRPP permutations (when \texttt{verbose=TRUE})
advanced.procD.lm

random.slope.dist
random pairwise distances between slopes from RRPP permutations (when verbose=TRUE)

random.slope.comp
random pairwise slope direction comparisons (r or angle) from RRPP permutations (when verbose=TRUE)

Author(s)

Michael Collyer

References


See Also

procD.lm

Examples

data(plethodon)
Y.gpa<-gpagen(plethodon$land) #GPA-alignment
Y<-two.d.array(Y.gpa$coords)
CS <- Y.gpa$Csize
sp<- plethodon$species
st<- plethodon$site

# Example of a nested model comparison (as with ANOVA with RRPP)
advanced.procD.lm(Y ~ log(CS) + sp, ~ log(CS)*sp*st, iter=19)

# Example of a test of a factor interaction, plus pairwise comparisons (replaces pairwiseD.test)
advanced.procD.lm(Y ~ st*sp, ~st + sp, groups = ~st*sp, iter=19)

# Example of a test of a factor interaction, plus pairwise comparisons, # accounting for a common allomtry (replaces pairwiseD.test)
advanced.procD.lm(Y ~ log(CS) + st*sp, ~log(CS) + st + sp, groups = ~st*sp, slope = ~log(CS), iter=19)

# Example of a test of homogeneity of slopes, plus pairwise slopes comparisons # (replaces pairwise.slope.test)
advanced.procD.lm(Y ~ log(CS)*st*sp, ~log(CS) + st*sp, groups = ~st*sp, slope = ~log(CS), angle.type = "deg", iter=19)

# Example of partial pairwise comparisons, given greater model complexity
advanced.procD.lm(Y ~ log(CS)*st*sp, ~log(CS) + st*sp, groups = ~sp, slope = ~log(CS), angle.type = "deg", iter=19)
arrayspecs  

Convert landmark data matrix into array \((p \times k \times n)\)

**Description**

Convert a matrix of landmark coordinates into a 3-dimensional array

**Usage**

arrayspecs(A, p, k)

**Arguments**

- A: A matrix containing landmark coordinates for a set of specimens
- p: Number of landmarks
- k: Number of dimensions (2 or 3)

**Details**

This function converts a matrix of landmark coordinates into a \((p \times k \times n)\) array, which is the required input format for many functions in geomorph. The input matrix can be arranged such that the coordinates of each landmark are found on a separate row, or that each row contains all landmark coordinates for a single specimen.

**Value**

Function returns a \((p \times k \times n)\) array, where \(p\) is the number of landmark points, \(k\) is the number of landmark dimensions (2 or 3), and \(n\) is the number of specimens. The third dimension of this array contains names for each specimen if specified in the original input matrix.

**Author(s)**

Dean Adams & Mike Collyer

**Examples**

```r
x <- matrix(rnorm(18), nrow=3) # Random triangles (all coordinates on same row for each triangle)
arrayspecs(x, 3, 2)

x2 <- matrix(rnorm(18), ncol=2) # Random triangles (each landmark on its own row)
arrayspecs(x2, 3, 2)
```
Analysis of bilateral symmetry

Description

Function performs an analysis of directional and fluctuating asymmetry for bilaterally symmetric objects.

Usage

```r
bilat.symmetry(A, ind = NULL, side = NULL, replicate = NULL,
               object.sym = FALSE, land.pairs = NULL, warpgrids = TRUE, mesh = NULL,
               verbose = FALSE, ShowPlot = TRUE)
```

Arguments

- **A**: An array \((p \times k \times n)\) containing GPA-aligned coordinates for a set of specimens [for "object.sym=FALSE, A is of dimension \((n \times k \times 2n)\)]
- **ind**: A vector containing labels for each individual. For matching symmetry, the matched pairs receive the same label (replicates also receive the same label).
- **side**: An optional vector (for matching symmetry) designating which object belongs to which ‘side-group’.
- **replicate**: An optional vector designating which objects belong to which group of replicates.
- **object.sym**: A logical value specifying whether the analysis should proceed based on object symmetry =TRUE or matching symmetry =FALSE.
- **land.pairs**: An optional matrix (for object symmetry) containing numbers for matched pairs of landmarks across the line of symmetry.
- **warpgrids**: A logical value indicating whether deformation grids for directional and fluctuating components of asymmetry.
- **mesh**: A mesh3d object to be warped to represent shape deformation of the directional and fluctuating components of asymmetry if warpgrids = TRUE (see warpRefMesh).
- **verbose**: A logical value indicating whether the output is basic or verbose (see Value below).
- **ShowPlot**: A logical value indicating whether or not the plot should be returned.

Details

The function quantifies components of shape variation for a set of specimens as described by their patterns of symmetry and asymmetry. Here, shape variation is decomposed into variation among individuals, variation among sides (directional asymmetry), and variation due to an individual x side interaction (fluctuating symmetry). These components are then statistically evaluated using Procrustes ANOVA and Goodall’s F tests (i.e. an isotropic model of shape variation). Methods for both matching symmetry and object symmetry can be implemented. Matching symmetry is
when each object contains mirrored pairs of structures (e.g., right and left hands) while object symmetry is when a single object is symmetric about a midline (e.g., right and left sides of human faces). Analytical and computational details concerning the analysis of symmetry in geometric morphometrics can be found in Mardia et al. 2000; Klingenberg et al. 2002.

Analyses of symmetry for matched pairs of objects is implemented when object.sym=FALSE. Here, a 3D array \([p \times k \times 2n]\) contains the landmark coordinates for all pairs of structures (2 structures for each of \(n\) specimens). Because the two sets of structures are on opposite sides, they represent mirror images, and one set must be reflected prior to the analysis to allow landmark correspondence. IT IS ASSUMED THAT THE USER HAS DONE THIS PRIOR TO PERFORMING THE SYMMETRY ANALYSIS. Reflecting a set of specimens may be accomplished by multiplying one coordinate dimension by '-1' for these structures (either the \(x\)-, the \(y\)-, or the \(z\)-dimension). A vector containing information on individuals and sides must also be supplied. Replicates of each specimen may also be included in the dataset, and when specified will be used as measurement error (see Klingenberg and McIntyre 1998).

Analyses of object symmetry is implemented when object.sym=TRUE. Here, a 3D array \([p \times k \times n]\) contains the landmark coordinates for all \(n\) specimens. To obtain information about asymmetry, the function generates a second set of objects by reflecting them about one of their coordinate axes. The landmarks across the line of symmetry are then relabeled to obtain landmark correspondence. The user must supply a list of landmark pairs. A vector containing information on individuals must also be supplied. Replicates of each specimen may also be included in the dataset, and when specified will be used as measurement error.

**Value**

Function returns a list with the following components:

- **ANOVA.shape**: Procrustes ANOVA table assessing patterns of shape asymmetry
- **ANOVA.size**: Procrustes ANOVA table assessing patterns of shape asymmetry (when object.sym=FALSE)
- **symm.shape**: The symmetric component of shape variation of the aligned specimens (when verbose=TRUE)
- **asymm.shape**: The asymmetric component of shape variation of the aligned specimens (when verbose=TRUE)

**Author(s)**

Dean Adams & Emma Sherratt

**References**


Examples

# Example of matching symmetry

data(mosquito)
bilat.symmetry(mosquito$wingshape, ind=mosquito$ind, side=mosquito$side, replicate=mosquito$replicate, object.sym=FALSE)

# Example of object symmetry

data(scallops)
bilat.symmetry(scallops$coorddata, ind=scallops$ind, object.sym=TRUE, land.pairs=scallops$land.pairs)

buildtemplate Build 3D surface template

Description

An interactive function to build template of three-dimensional (3D) surface sliding semilandmarks. Input for the function is either a matrix of vertex coordinates defining a 3D surface object or a mesh3d object as obtained from read.ply.

Usage

buildtemplate(spec, fixed, surface.sliders, ptsize = 1, center = TRUE)

Arguments

spec Name of surface file, as either an object of class shape3d/mesh3d, or matrix of three-dimensional vertex coordinates.
fixed numeric Either: a single value designating the number of fixed template landmarks to be selected by digit.fixed, OR a p-x-k matrix of 3D coordinates collected previously
surface.sliders numeric: The number of template surface sliders desired
ptsizes numeric: Size to plot the mesh points (vertices), e.g. 0.1 for dense meshes, 3 for sparse meshes
center Logical Whether the object 'spec' should be centered prior to digitizing (default center=TRUE)

Details

Function constructs a template of fixed landmarks and n "surface sliders", semilandmarks that slide over a surface. The user digitizes the fixed points (see digitizing below), then the function finds n surface semilandmarks following algorithm outlined in Gunz et al. (2005) and Mitteroecker and Gunz (2009). Surface semilandmarks are roughly equidistant set of predetermined number of points, chosen over the mesh automatically using a nearest-neighbor approach. The set of fixed
and surface slider landmarks are exported as a "template", which is used to extract a set of similarly numbered landmarks on every specimen using function `digitsurface`. Some of the "fixed" landmarks can be later designated as "curve sliders" using function `define.sliders` if required - see details in `digit.fixed`. Because template matching is based on the correspondence of fixed landmark points in the template and the specimen, a minimum of four fixed landmarks must be used.

To ensure a strong match between the scan and the template, it is recommended that a reasonable number of fixed points be used. These fixed points can be designated as "curve sliders" later using function `define.sliders`, see the function `digit.fixed` for details. NOTE: Function centers the mesh before digitizing by default (center=TRUE). If one chooses not to center, specimen may be difficult to manipulate in rgl window.

**Digitizing:**  Digitizing using `buildtemplate` is interactive between landmark selection using a mouse (see below for instructions), and the R console. Once a point is selected, the user is asked if the system should keep or discard the selection #'(y/n). If "y", the user is asked to continue to select the next landmark. If "n" the removes the last chosen landmark, and the user is asked to select it again. This can be repeated until the user is comfortable with the landmark chosen.

To digitize with a standard 3-button (PC):

1. the RIGHT mouse button (primary) to select points to be digitized,
2. the LEFT mouse button (secondary) is used to rotate mesh,
3. the mouse SCROLLER (third/middle) is used to zoom in and out.

NOTE: Digitizing functions on MACINTOSH computers using a standard 3-button mice works as specified. Macs using platform specific single button mice, XQuartz must be configured: go to Preferences > Input > tick "Emulate three button mouse":

1. press button to rotate 3D mesh,
2. press button while pressing COMMAND key to select vertex to be used as a landmark),
3. press button while pressing OPTION key to adjust mesh perspective.
4. the mouse SCROLLER or trackpad two finger scroll is used to zoom in an out.

NOTE: there is no pan (translate) functionality in rgl library for all platforms at this time. The template can be edited using function `edittemplate`.

**AUTO mode:** The function as described above (for interactive mode) calls `digit.fixed`, prompting the user to select fixed landmarks in the rgl window. However if the user has digitized these fixed landmark elsewhere (e.g., in other software), then the input for parameter 'fixed' can be a p-x-k matrix of 3D coordinates. In this case, the function will automatically use these landmarks to build the template of sliding semilandmarks.

**Value**

Function writes to the working directory three files: an NTS file with the name of the specimen and .nts suffix containing the landmark coordinates, "template.txt" containing the same coordinates for use with the function `digitsurface`, and "surfslide.csv", a file containing the address of the landmarks defined as "surface sliders" for use with `gpagen`. Function also returns to console an n x 3 matrix containing the x,y,z coordinates of the digitized landmarks.

**Author(s)**

Erik Otarola-Castillo & Emma Sherratt
References


See Also

read.ply
digit.fixed
digitsurface

---

**compare.evol.rates**

Comparing rates of shape evolution on phylogenies

**Description**

Function calculates rates of shape evolution for two or more groups of species on a phylogeny from a set of Procrustes-aligned specimens.

**Usage**

```r
compare.evol.rates(phy, A, gp, ShowPlot = TRUE, iter = 999)
```

**Arguments**

- `phy` A phylogenetic tree of class phylo - see `read.tree` in library ape
- `A` A matrix (n x [p x k]) or 3D array (p x k x n) containing GPA-aligned coordinates for a set of specimens
- `gp` A factor array designating group membership
- `ShowPlot` A logical value indicating whether or not the plot should be returned
- `iter` Number of iterations for significance testing

**Details**

The function compares rates of morphological evolution for two or more groups of species on a phylogeny, under a Brownian motion model of evolution. It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with `gpagen`]. The approach is based on the distances between species in morphospace after phylogenetic transformation (Adams 2014). From the data the rate of shape evolution for each group is calculated, and a ratio of rates is obtained. If three or more groups of species are used, the ratio of the maximum to minimum rate is used as a test statistic (see Adams 2014). Significance testing is accomplished by phylogenetic simulation in which tips data are obtained under Brownian motion using a common evolutionary rate pattern for all species on the phylogeny (i.e., a common evolutionary rate matrix). This procedure is more general, and retains the desirable statistical properties of earlier methods,
but for a wider array of data types. If three or more groups of species are used, pairwise p-values are also returned. A histogram of evolutionary rate ratios obtained via phylogenetic simulation is presented, with the observed value designated by an arrow in the plot. The function can be used to obtain a rate for the whole dataset of species by using a dummy group factor assigning all species to one group.

This function can be used with univariate data (i.e. centroid size) if imported as matrix with row-names giving the taxa names.

**Value**

Function returns a list with the following components:

- `sigma.d`: The phylogenetic evolutionary rate for all species on the phylogeny
- `sigmad.all`: The phylogenetic evolutionary rate for each group of species on the phylogeny
- `sigmad.ratio`: The ratio of maximum to minimum evolutionary rates
- `pvalue`: The significance level of the observed ratio
- `pairwise.pvalue`: Matrix of pairwise significance levels comparing each pair of rates

**Author(s)**

Dean Adams & Emma Sherratt

**References**


**Examples**

```r
data(plethspecies)
Y.gpa<-gpagen(plethspecies$land) #GPA-alignment

gp.end<-factor(c(0,0,1,0,0,1,0,0)) #endangered species vs. rest
names(gp.end)<-plethspecies$phy$tip

#Calculate rates of shape
compare.evolverates(plethspecies$phy,Y.gpa$coords,gp=gp.end,iter=49)

#Calculate rates of size
compare.evolverates(plethspecies$phy,Y.gpa$Csize,gp=gp.end,iter=49)
```
**compare.modular.partitions**

*Compare modular signal to alternative landmark subsets*

**Description**

Function quantifies the degree of modularity between two or more hypothesized modules of Procrustes-aligned landmark coordinates and compares this to patterns found by randomly assigning landmarks into subsets.

**Usage**

```r
compare.modular.partitions(A, partition.gp, ShowPlot = TRUE, iter = 999)
```

**Arguments**

- `A` A 3D array (p x k x n) containing GPA-aligned coordinates for all specimens, or a matrix (n x variables)
- `partition.gp` A list of which landmarks (or variables) belong in which partition (e.g. A,A,A,B,B,B,C,C,C)
- `ShowPlot` A logical value indicating whether or not the plot should be returned
- `iter` Number of iterations for significance testing

**Details**

The function quantifies the degree of modularity in two or more hypothesized modules of shape data as defined by landmark coordinates, and compares this to the degree of modular signal found in random assignment of landmarks to modules. It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with `gpagen`]. The degree of modularity is quantified using the RV coefficient (Klingenberg 2009). If more than two modules are defined, the average RV coefficient is utilized (see Klingenberg 2009). The RV coefficient for the observed modular hypothesis is then compared to a distribution of values obtained by randomly assigning landmarks into subsets, with the restriction that the number of landmarks in each subset is identical to that observed in each of the original partitions. A significant modular signal is found when the observed RV coefficient is small relative to this distribution (see Klingenberg 2009). A histogram of coefficients obtained via resampling is presented, with the observed value designated by an arrow in the plot.

Landmark groups can be defined using `define.modules`, or made by hand (see example below). To use this method with other data (i.e., a set of length measurements), the input `A` should be a matrix of n rows of specimens and variables arranged in columns. In this case, the `partition.gp` input should have each variable assigned to a partition.

**Value**

Function returns a list with the following components:

- `RV` The estimate of the observed modular signal
compare.multi.evol.rates

pvalue
 The significance level of the observed signal

RV.min
 The minimal RV coefficient found via landmark permutation

RV.min.partitions
 A list of landmarks assigned to partitions that yields the minimal RV coefficient

Author(s)

Dean Adams

References


See Also

define.modules

Examples

data(plethodon)
Y.gpa<-gpagen(plethodon$land)   #GPA-alignment
#landmarks on the skull and mandible assigned to partitions
land.gps<-c("A","A","A","A","A","B","B","B","B","B","B","B")

compare.modular.partitions(Y.gpa$coords,land.gps,iter=99)
#Result implies that the skull and mandible are not independent modules

compare.multi.evol.rates

Comparing rates of shape evolution among traits on phylogenies

Description

Function calculates rates of shape evolution for two or more multi-dimensional traits on a phylogeny from a set of Procrustes-aligned specimens

Usage

compare.multi.evol.rates(A, gp, phy, Subset = TRUE, ShowPlot = TRUE, iter = 999)
**Arguments**

- **A**: A matrix \((n \times [p \times k])\) or 3D array \((p \times k \times n)\) containing GPA-aligned coordinates for a set of specimens.
- **gp**: A factor array designating group membership.
- **phy**: A phylogenetic tree of class **phylo** - see `read.tree` in library **ape**.
- **Subset**: A logical value indicating whether or not the traits are subsets from a single landmark configuration (default is TRUE).
- **ShowPlot**: A logical value indicating whether or not the plot should be returned.
- **iter**: Number of iterations for significance testing.

**Details**

The function compares rates of morphological evolution for two or more multi-dimensional traits on a phylogeny, under a Brownian motion model of evolution following the procedure of Denton and Adams (2015). It is assumed that the landmarks for all traits have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with `gpagen`]. The approach calculates multivariate evolutionary rates found from the distances between species in morphospace after phylogenetic transformation (sensu Adams 2014). From the data the rate of shape evolution for each multi-dimensional trait is calculated, and a ratio of rates is obtained. If three or more traits are used, the ratio of the maximum to minimum rate is used as a test statistic (see Denton and Adams 2015). Significance testing is accomplished by phylogenetic simulation in which tips data are obtained under Brownian motion using a an evolutionary rate matrix for all traits, which contains a common rate for all trait dimensions (Denton and Adams 2015). If three or more traits are used, pairwise p-values are also returned. A histogram of evolutionary rate ratios obtained via phylogenetic simulation is presented, with the observed value designated by an arrow in the plot.

The shape data may be input as either a 3D array \((p \times k \times n)\) containing GPA-aligned coordinates for a set of species, or as a matrix \((n \times [p \times k])\) whose rows correspond to each species. In both cases, species names must be provided as rownames (for a matrix) or as the names of the third dimension of the array. Landmark groups for each trait are then specified by a factor array designating which landmark belongs to which trait. Additionally, if the method is to be used with other data (i.e., a set of length measurements), the input A should be a matrix of \(n\) rows of species and \(p\) columns of variables. In this case, the grouping factor should have each variable assigned to a trait group.

Comparisons of evolutionary rates between traits may be accomplished in one of two ways. First, if the traits are are part of a single shape that was subjected to a single Procrustes superimposition (i.e., they are subsets of landmarks in the configuration), then the procedure is performed without alteration as described above. However, if the shapes are derived from different structures (shapes) that were superimposed separately, then the estimates of the rates must take the difference in the number of trait dimensions into account (see discussion in Denton and Adams 2015). This option is identified by selecting Subset = FALSE.

**Value**

Function returns a list with the following components:

- **rates.all**: The phylogenetic evolutionary rates for each trait.
- **rate.ratio**: The ratio of maximum to minimum evolutionary rates.
define.links

pvalue The significance level of the observed rate ratio
pvalue.gps Matrix of pairwise significance levels comparing each pair of rates

Author(s)

Dean Adams

References


Examples

data(plethspecies)
yNgpa<gpagen(plethspecies$land)  #GPA-alignment
land.gp<(-c("A","A","A","A","A","B","B","B","B","B")  #mandible and cranium subsets

compare.multi.evol.rates(yNgpa$coords,land.gp,plethspecies$phy,iter=99)

define.links

Define links between landmarks

Description

An interactive function to define which landmarks should be linked to aid visualization.

Usage

define.links(spec, ptsize = 1, links = NULL)

Arguments

spec Name of specimen, as an object matrix containing 2D or 3D landmark coordinates
ptsize Numeric Size to plot the landmarks
links Optional An existing links matrix to add on to
**define.modules**

**Details**

Function takes a matrix of digitized landmark coordinates (e.g. from mshape) and allows the user to define pairs of landmarks to be linked, for visualization purposes. The output is a matrix to be used by plotAllSpecimens & plotReftoTarget option 'links='.

**Selection:** In the plot window select two landmarks that will be linked. In the console, the user will be prompted to continue linking landmarks to build the wireframe or end the session (typing y or n respectively). For 2D data in plot window, use LEFT mouse button to select landmarks. For 3D data in rgl window, use RIGHT mouse button (or command+LEFT for mac) to select landmarks.

**Value**

Function returns a matrix of which landmarks will be linkes to be used by plotAllSpecimens & plotReftoTarget option 'links='.

**Author(s)**

Emma Sherratt

**See Also**

plotAllSpecimens
plotReftoTarget

---

**define.modules**  
*Define modules (landmark partitions)*

**Description**

An interactive function to define which landmarks should be assigned to each module (landmark partition).

**Usage**

```r
define.modules(spec, nmodules)
```

**Arguments**

- **spec**  
  Name of specimen, as an object matrix containing 2D or 3D landmark coordinates

- **nmodules**  
  Number of modules to be defined
Details

Function takes a matrix of digitized landmark coordinates (e.g. from mshape) and allows the user to assign landmarks to each module. The output is a list of which landmarks belong in which partition, to be used by compare.modular.partitions.

Selection in 2D: Choosing which landmarks will be included in each module involves landmark selection using a mouse in the plot window. The user is prompted to select each landmark in turn to be assigned to module 1: using the LEFT mouse button (or regular button for Mac users), click on the hollow circle to choose the landmark. Selected landmarks will be filled in. When all landmarks for module 1 are chosen, press 'esc', and then start selecting landmarks for module 2. Repeat until all modules are defined.

Selection in 3D: Choosing which landmarks will be included in each module involves landmark selection using a mouse in the rgl plot window. The user is prompted to select one or more landmarks. To do so, use the RIGHT mouse button (or command + LEFT button for Mac users), draw a rectangle around landmarks to select. Selected landmarks will be coloured yellow. Then type into the console a letter (e.g. 1, 2, 3...) to assign selected landmark(s) to this module. Repeat until all landmarks are assigned to modules.

Value

Function returns a vector of which landmarks belong in which module (e.g. 1,1,2,2,3,3,3,2) to be used with compare.modular.partitions option 'landgroups'.

Author(s)

Emma Sherratt

See Also

compare.modular.partitions

define.sliders

Select points to "slide" along curves.

Description

An interactive function to define which landmarks will "slide" along two-dimensional (2D) or three-dimensional (3D) curves.

Usage

define.sliders(landmarks, nsiders, surfsliders = NULL, write.file = TRUE)
define.sliders

Arguments

landmarks A matrix containing 2D or 3D landmark coordinates of landmarks and semilandmarks, OR A vector containing a sequence of numbers corresponding to the landmarks in the order they appear along the curve (for AUTO mode)

nsliders Number of landmarks to be semilandmarks that slide along curves

surfsliders (3D only) If 'landmarks' contains "surface sliders", e.g. made by buildtemplate, these should be given as a vector or use surfsliders = T, and function looks for "surfslide.csv" in working directory.

write.file A logical value indicating whether the matrix is written to file as .csv.

Details

Function takes a matrix of digitized landmark coordinates, such as made by digitize2d or digit.fixed, and helps user choose which landmarks will be treated as "sliders" in Generalized Procrustes analysis gpagen. This type of semilandmark "slides" along curves lacking known landmarks (see Bookstein 1997 for algorithm details). Each sliding semilandmark ("sliders") will slide between two designated points, along a line tangent to the specified curvature.

Defining landmarks is an interactive procedure (see below for 2D and 3D routines). The procedure is overlapping. For example: there are 5 landmarks (1:5), 1 and 5 are landmarks and 2,3,4 are sliders, the user must select '1' '2' '3', and then '2' '3' '4', and then '3' '4' '5'.

Selection in 2D: Choosing which landmarks will be sliders involves landmark selection using a mouse in the plot window. To define the sliders, for each sliding landmark along the curve in the format 'before-slider-after', using the LEFT mouse button (or regular button for Mac users), click on the hollow circle to choose the landmark in the following order:

1. Click to choose the first landmark between which semi-landmark will "slide",
2. Click to choose sliding landmark,
3. Click to choose the last landmark between which semi-landmark will "slide". Selected landmarks will be filled in and lines are drawn connecting the three landmarks, and will highlight the sliding semilandmark in red and the flanking landmarks in blue.

Selection in 3D: Choosing which landmarks will be sliders involves landmark selection using a mouse in the rgl plot window. With a standard 3-button (PC) buildtemplate uses:

1. the RIGHT mouse button (primary) to choose points to be defined as sliders,
2. the LEFT mouse button (secondary) is used to rotate mesh,
3. the mouse SCROLLER (third/middle) is used to zoom in and out.

NOTE: Digitizing functions on MACINTOSH computers using a standard 3-button mouse works as specified. Macs using platform specific single button mice, XQuartz must be configured: go to Preferences > Input > tick "Emulate three button mouse":

1. press button to rotate 3D mesh,
2. press button while pressing COMMAND key to select points to be defined as sliders,
3. press button while pressing OPTION key to adjust mesh perspective.
4. the mouse SCROLLER or trackpad two finger scroll is used to zoom in an out.

To define the sliders, for each sliding landmark along the curve in the format 'before-slider-after':

1. Click to choose the first landmark between which semi-landmark will "slide",
2. Click to choose sliding landmark,
3. Click to choose the last landmark between which semi-landmark will "slide". Screen will show lines connecting the three landmarks, and will highlight the sliding semilandmark in red.

**AUTO mode:** The input 'landmarks' can be simply a vector of numbers corresponding to the "sliders" (semilandmarks) in the order they appear along a curve on the specimen. This can be made by c() or seq() or any other reasonable method.

If the sliders form a closed curve, then the function assumes that the first and last landmarks in the 'landmarks' vector are THE SAME are fixed (not sliders). e.g. if landmark 1 is a fixed landmark, and 2, 3 and 4 are semilandmarks, then sliders = c(1,2,3,4,1).

**Value**

Function returns a 'nsliders-x-3' matrix containing the landmark address of the curve sliders, indicating the landmarks between which the slider landmarks will "slide". If write.file = T the matrix is also written to working directory as "curveslide.csv". Matrix (or "curveslide.csv") is designed for use by gpagen during GPA.

**Author(s)**

Emma Sherratt, Dean Adams, Erik Otarola-Castillo

**References**


**See Also**

digitize2d, digit.fixed, gpagen, digit.curves

digit.curves

**Calculate semilandmarks along a curve**

**Description**

A function to calculate equidistant two-dimensional and three-dimensional semilandmarks along a curve. These landmarks will be treated as "sliders" in Generalized Procrustes analysis gpagen. This type of semilandmark "slides" along curves lacking known landmarks (see Bookstein 1997 for algorithm details). Each sliding semilandmark ("sliders") will slide between two designated points, along a line tangent to the specified curvature, as specified by define.sliders.

**Usage**

digit.curves(start, curve, nPoints, closed = T)
digit.curves

Arguments

start       A vector of x,y,(z) coordinates for the fixed landmark defining the start of the curve
curve       A p-x-k matrix of 2D or 3D coordinates for a set of ordered points defining a curve
nPoints     Numeric how many semilandmarks to place equidistantly along the curve
closed      Logical Whether the curve is closed (TRUE) or open (FALSE)

Details

The function is based upon tpsDig2 'resample curve by length' for 2D data by James Rohlf. The start of the curve is a fixed landmark on the curve that is equivalent (homologous) in each specimen in the sample (and will be treated as a fixed point during Procrustes Superimpoistion using gpagen). Then nPoints are calculated along the curve at equidistant points from the start to the end.

'curve' is a p-x-k matrix of 2D or 3D coordinates for a set of ordered points defining a curve. This can be the pixels of an outline calculated in ImageJ (save xy coordinates) or any other reasonable way of obtaining ordered coordinates along a curve (including sampling by hand using digit.fixed or digitize2d - but note that there should be more points defining the curve than nPoints in order to accurately calculate the semilandmarks).

If 'closed = T', the function returns the coordinates of the 'start' landmark plus nPoints. If 'closed = F', the function returns the coordinates of the 'start' landmark, plus nPoints and the end of the curve.

Value

Function returns a matrix of coordinates for nPoints equally spaced semilandmarks sampled along the curve (plus start and end if 'closed = F', or only including start if 'closed = T')

Author(s)

Emma Sherratt

References


See Also

digit.fixed digitize2d
digit.fixed

**Digitize 3D landmarks**

**Description**

An interactive function to digitize three-dimensional (3D) landmarks. Input for the function is either a matrix of vertex coordinates defining a 3D surface object or a mesh3d object as obtained from `read.ply`.

**Usage**

```r
digit.fixed(spec, fixed, index = FALSE, ptsize = 1, center = TRUE)
```

**Arguments**

- `spec`: An object of class shape3d/mesh3d, or matrix of 3D vertex coordinates
- `fixed`: Numeric The number landmarks (fixed, and curve sliders if desired)
- `index`: Logical Whether selected landmark addresses should be returned (internal use only)
- `ptsizes`: Numeric Size to plot the mesh points (vertices), e.g. 0.1 for dense meshes, 3 for sparse meshes
- `center`: Logical Whether the object 'spec' should be centered prior to digitizing (default center=TRUE)

**Details**

Function for digitizing "n" three-dimensional landmarks. The landmarks are "fixed" (traditional landmarks). They can be later designated as "curve sliders" (semilandmarks, that will "slide" along curves lacking known landmarks if required. A sliding semi-landmark ("sliders") will slide between two designated points, along a line tangent to the specified curvature, and must be defined as "sliders" using function `define.sliders` or with similar format matrix made outside R.

For 3D "surface sliders" (surface semilandmarks that slide over a surface) the function `digitsurface` should be used instead. NOTE: Function centers the mesh before digitizing by default (center=TRUE). If one chooses not to center, specimen may be difficult to manipulate in rgl window.

**Digitizing**: Digitizing is interactive between landmark selection using a mouse (see below for instructions), and the R console. Once a point is selected, the user is asked if the system should keep or discard the selection #(y/n). If "y", the user is asked to continue to select the next landmark. If "n" the removes the last chosen landmark, and the user is asked to select it again. This can be repeated until the user is comfortable with the landmark chosen.

To digitize with a standard 3-button (PC):

1. the RIGHT mouse button (primary) to select points to be digitized,
2. the LEFT mouse button (secondary) is used to rotate mesh,
3. the mouse SCROLLER (third/middle) is used to zoom in and out.
NOTE: Digitizing functions on MACINTOSH computers using a standard 3-button mice works as specified. Macs using platform specific single button mice, XQuartz must be configured: go to Preferences > Input > tick "Emulate three button mouse":

1. press button to rotate 3D mesh,
2. press button while pressing COMMAND key to select vertex to be used as a landmark,
3. press button while pressing OPTION key to adjust mesh perspective.
4. the mouse SCROLLER or trackpad two finger scroll is used to zoom in an out.

NOTE: there is no pan (translate) functionality in rgl library for all platforms at this time.

Value

Function writes to the working directory an NTS file with the name of the specimen and .nts suffix containing the landmark coordinates. If index=FALSE function returns to the console an n x 3 matrix containing the x,y,z coordinates of the digitized landmarks. If index=TRUE, function returns a list:

- selected: a matrix containing the x,y,z coordinates of the digitized landmarks
- fix: a matrix of addresses for landmarks that are "fixed" (for internal use)

Author(s)

Erik Otarola-Castillo & Emma Sherratt

See Also

read.ply

digitize2d

Description

An interactive function to digitize two-dimensional(2D) landmarks from .jpg files.

Usage

digitize2d(filelist, nlandmarks, scale = NULL, tpsfile, verbose = TRUE)

Arguments

- filelist: A list of names of jpeg images to be digitized.
- nlandmarks: Number of landmarks to be digitized.
- scale: An vector containing the length of the scale to be placed on each image.
- tpsfile: The name of a TPS file to be created or read.
- verbose: logical. User decides whether to digitize in verbose or silent format (see details), default is TRUE.
Details

This function may be used for digitizing 2D landmarks from jpeg images (.jpg). The user provides a list of image names, the number of landmarks to be digitized, and the name of an output TPS file. An option is included to allow the user to digitize a scale on each image to convert the landmark coordinates from pixels into meaningful units. Landmarks to be digitized can include both fixed landmarks and semi-landmarks, the latter of which are to be designated as "sliders" for subsequent analysis (see the function \texttt{define.sliders}).

**The Digitizing Session:** Digitizing landmarks from 2D photos requires that a scale bar is placed in the image in order to scale the coordinate data. The 'scale' option requires: a single number (e.g. 10) which means that the scale to be measured in all images is a 10mm scale bar; OR a vector the same length as the filelist containing a number for the scale of each image. If scale=NULL, then the digitized coordinates will not be scaled. This option is NOT recommended.

Users may digitize all specimens in one session, or may return at a later time to complete digitizing. In the latter case, the user provides the same filelist and TPS file and the function will determine where the user left off.

If specimens have missing landmarks, these can be incorporated during the digitizing process using the 'a' option as described below (a=absent).

**Specimen Digitizing:** Digitizing landmarks involves landmark selection using a mouse in the plot window, using the LEFT mouse button (or regular button for Mac users):

1. Digitize the scale bar (if requested) by selecting the two end points. Use a single click for start and end points. The user is asked whether the system should keep or discard the digitized scale bar.

2. Digitize each landmark with single click and the landmark is shown in red.

If verbose = TRUE, digitizing is interactive between landmark selection using a mouse and the R console. Once a landmark is selected, the user is asked if the system should keep or discard the selection (y/n/a). If "y", the user is asked to continue to select the next landmark. If "n", the user is asked to select it again.

To digitize a missing landmark, simply click on any location in the image. Then, when prompted to keep selection, choose 'a' (for absent). Missing landmarks can only be included during the digitizing process when verbose=TRUE.

If verbose = FALSE the digitizing of landmarks is continuous and uninterrupted. Here the user will not be prompted to approve each landmark selection.

At the end of digitizing, the landmark coordinates are written to a TPS file. The x,y values are unscaled if a vector of scales is included, and the scale is returned on line SCALE= after each specimen x,y data.

**Value**

Function returns a tps file containing the digitized landmark coordinates.

**Author(s)**

Dean Adams, Erik Otarola-Castillo and Emma Sherratt
digitsurface

Digitize 3D fixed landmarks and surface semilandmarks.

Description

An interactive function to digitize three-dimensional (3D) landmarks on a surface lacking known landmarks. Input for the function is either a matrix of vertex coordinates defining a 3D surface object or a mesh3d object as obtained from read.ply.

Usage

digitsurface(spec, fixed, ptsize = 1, center = TRUE)

Arguments

- **spec**: Name of surface file, as either an object of class shape3d/mesh3d, or matrix of three-dimensional vertex coordinates.
- **fixed**: numeric Either: a single value designating the number of fixed template landmarks to be selected by digit.fixed, OR a p-x-k matrix of 3D coordinates collected previously
- **ptsizesize**: numeric: Size to plot the mesh points (vertices), e.g. 0.1 for dense meshes, 3 for sparse meshes
- **center**: Logical: Whether the object 'spec' should be centered prior to digitizing (default center=TRUE)

Details

Function for digitizing fixed 3D landmarks and placing "surface sliders", semilandmarks that slide over a surface. Following selection of fixed points (see digitizing below), function finds surface semilandmarks following algorithm outlined in Gunz et al. (2005) and Mitteroecker and Gunz (2009). digitsurface finds the same number of surface semilandmarks as the template (created by buildtemplate) by downsampling scanned mesh, registering template with current specimen via GPA. A nearest neighbor algorithm is used to match template surface landmarks to current specimen's. To use function digitsurface, the template must be constructed first, and 'template.txt' be in the working directory. Because template matching is based on the correspondence of fixed landmark points in the template and the specimen, a minimum of four fixed landmarks must be used.

Some of the "fixed" landmarks digitized with digitsurface can be later designated as "curve sliders" using function define.sliders if required (see details in digit.fixed). NOTE: Function centers the mesh before digitizing by default (center=TRUE). If one chooses not to center, specimen may be difficult to manipulate in rgl window.

**Digitizing:** Digitizing is interactive between landmark selection using a mouse (see below for instructions), and the R console. Once a point is selected, the user is asked if the system should keep or discard the selection #'(y/n). If "y", the user is asked to continue to select the next
landmark. If "n" the removes the last chosen landmark, and the user is asked to select it again. This can be repeated until the user is comfortable with the landmark chosen.

To digitize with a standard 3-button (PC):

1. the RIGHT mouse button (primary) to select points to be digitized,
2. the LEFT mouse button (secondary) is used to rotate mesh,
3. the mouse SCROLLER (third/middle) is used to zoom in and out.

NOTE: Digitizing functions on MACINTOSH computers using a standard 3-button mice works as specified. Macs using platform specific single button mice, XQuartz must be configured: go to Preferences > Input > tick "Emulate three button mouse":

1. press button to rotate 3D mesh,
2. press button while pressing COMMAND key to select vertex to be used as a landmark,
3. press button while pressing OPTION key to adjust mesh perspective.
4. the mouse SCROLLER or trackpad two finger scroll is used to zoom in and out.

NOTE: there is no pan (translate) functionality in rgl library for all platforms at this time.

AUTO mode: The function as described above (for interactive mode) calls digit.fixed, prompting the user to select fixed landmarks in the rgl window. However if the user has digitized these fixed landmark elsewhere (e.g., in other software), then the input for parameter 'fixed' can be a p-x-k matrix of 3D coordinates. In this case, the function will automatically use these landmarks and fit the template of sliding semilandmarks.

Value

Function writes to the working directory an NTS file with the name of the specimen and .nts suffix containing the landmark coordinates.

Author(s)

Erik Otarola-Castillo & Emma Sherratt

References


See Also

buildtemplate
read.ply
digit.fixed
**editTemplate**

*Edit 3D template*

**Description**

An interactive function to remove landmarks from a 3D template file.

**Usage**

```r
editTemplate(template, fixed, n)
```

**Arguments**

- `template` Matrix of template 3D coordinates.
- `fixed` Number of "fixed" landmark points (non surface sliding points)
- `n` Number of points to be removed

**Details**

Function edits a 'template.txt' file made by `buildtemplate`, which must be in current working directory. Function overwrites 'template.txt' in working directory with edited version. Use `read.table("template.txt", header = T)`.

**Selection:** Choosing which landmarks will be deleted involves landmark selection using a mouse in the rgl plot window. With a standard 3-button (PC) `buildtemplate` uses:

1. the RIGHT mouse button (primary) to choose points to be deleted,
2. the LEFT mouse button (secondary) is used to rotate mesh,
3. the mouse SCROLLER (third/middle) is used to zoom in and out.

NOTE: Digitizing functions on MACINTOSH computers using a standard 3-button mice works as specified. Macs using platform specific single button mice, XQuartz must be configured: go to Preferences > Input > tick "Emulate three button mouse":

1. press button to rotate 3D mesh,
2. press button while pressing COMMAND key to select points to be deleted,
3. press button while pressing OPTION key to adjust mesh perspective.
4. the mouse SCROLLER or trackpad two finger scroll is used to zoom in and out.

**Value**

Function returns a matrix containing the x,y,z coordinates of the new template landmarks. Function also writes to working directory 'template.txt' containing the x,y,z coordinates of the template.

**Author(s)**

Erik Otarola-Castillo & Emma Sherratt
estimate.missing

Estimate locations of missing landmarks

Description

A function for estimating the locations of missing landmarks

Usage

estimate.missing(A, method = c("TPS", "Reg"))

Arguments

A
An array (p x k x n) containing landmark coordinates for a set of specimens

method
Method for estimating missing landmark locations

Details

The function estimates the locations of missing landmarks for incomplete specimens in a set of landmark configurations, where missing landmarks in the incomplete specimens are designated by NA in place of the x,y,z coordinates. Two distinct approaches are implemented.

The first approach (method="TPS") uses the thin-plate spline to interpolate landmarks on a reference specimen to estimate the locations of missing landmarks on a target specimen. Here, a reference specimen is obtained from the set of specimens for which all landmarks are present. Next, each incomplete specimen is aligned to the reference using the set of landmarks common to both. Finally, the thin-plate spline is used to estimate the locations of the missing landmarks in the target specimen (Gunz et al. 2009).

The second approach (method="Reg") is multivariate regression. Here each landmark with missing values is regressed on all other landmarks for the set of complete specimens, and the missing landmark values are then predicted by this linear regression model. Because the number of variables can exceed the number of specimens, the regression is implemented on scores along the first set of PLS axes for the complete and incomplete blocks of landmarks (see Gunz et al. 2009).

One can also exploit bilateral symmetry to estimate the locations of missing landmarks. Several possibilities exist for implementing this approach (see Gunz et al. 2009). Example R code for one implementation is found in Claude (2008).

NOTE: Because all geometric morphometric analyses and plotting functions implemented in geomorph require a full complement of landmark coordinates, the alternative to estimating the missing landmark coordinates is to proceed with subsequent analyses EXCLUDING specimens with missing values. To do this, see functions complete.cases (use: mydata[complete.cases(mydata),]) or na.omit (use: newdata <- na.omit(mydata)) to make a dataset of only the complete specimens. These functions require the dataset to be a matrix in the form of a 2d array (see two.d.array).

Value

Function returns an array (p x k x n) of the same dimensions as input A, including coordinates for the target specimens (the original landmarks plus the estimated coordinates for the missing landmarks). These data need to be Procrustes Superimposed prior to analysis (seesee gpagen).
findMeanSpec

Author(s)
Dean Adams

References

Examples
```r
data(plethodon)
plethland<-plethodon$land
plethland[3,2]<-plethland[8,2]<-NA  # create missing landmarks
plethland[3,10]<-NA

estimate.missing(plethland,method="TPS")
estimate.missing(plethland,method="Reg")
```

---

| findMeanSpec | Identify specimen closest to the mean of a set of aligned specimens |

Description
A function to identify which specimen lies closest to the estimated mean shape for a set of aligned specimens.

Usage
```r
findMeanSpec(A)
```

Arguments
- `A` A 3D array (p x k x n) containing landmark coordinates for a set of aligned specimens

Details
Function takes an array of aligned specimens (such as made by `gpagen`), calculates the distance of each to the estimated mean shape, and returns the name and address of the closest specimen. This function can be used to identify the specimen to be used by `warpRefMesh`. 
Value

Function returns the name and address of the specimen closest to the mean of the set of aligned specimens.

Author(s)

Emma Sherratt

See Also

warpRefMesh

---

fixed.angle  
*Rotate a subset of 2D landmarks to common articulation angle*

Description

A function for rotating a subset of landmarks so that the articulation angle between subsets is constant

Usage

```r
fixed.angle(A, art.pt = NULL, angle.pts = NULL, rot.pts = NULL, angle = 0, degrees = FALSE)
```

Arguments

- `A`: An array (p x k x n) containing landmark coordinates for a set of specimens
- `art.pt`: A number specifying which landmark is the articulation point between the two landmark subsets
- `angle.pts`: A vector containing numbers specifying which two points used to define the angle (one per subset)
- `rot.pts`: A vector containing numbers specifying which landmarks are in the subset to be rotated
- `angle`: An optional value specifying the additional amount by which the rotation should be augmented (in radians)
- `degrees`: A logical value specifying whether the additional rotation angle is expressed in degrees or radians (radians is default)

Details

This function standardizes the angle between two subsets of landmarks for a set of specimens. The approach assumes a simple hinge-point articulation between the two subsets, and rotates all specimens such that the angle between landmark subsets is equal across specimens (see Adams 1999). As a default, the mean angle is used, though the user may specify an additional amount by which this may be augmented.

Presently, the function is only implemented for two-dimensional landmark data.
Value

Function returns a (p x k x n) array of landmark coordinates.

Author(s)

Dean Adams

References


Examples

# Example using Plethodon
# Articulation point is landmark 1, rotate mandibular landmarks (2-5) relative to cranium

data(plethspecies)
fixed.angle(plethspecies$land,art.pt=1,angle.pts=c(5,6),rot.pts=c(2,3,4,5))

Description

Function quantifies the overall level of morphological integration for a set of Procrustes-aligned coordinates

Usage

globalIntegration(A, ShowPlot = TRUE)

Arguments

A 3D array (p1 x k x n) containing GPA-aligned coordinates
ShowPlot A logical value indicating whether or not the plot should be returned

Details

The function quantifies the overall level of morphological integration for a set of Procrustes coordinates. It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with gpagen]. Based on the set of aligned specimens, the function estimates the set of bending energies at various spatial scales, and plots the log of the variance of the partial warps versus the log of their corresponding bending energies (Bookstein 2015). For slope of a regression of these data provides information regarding the degree of overall morphological integration (or lack thereof).

A slope of negative one corresponds to self-similarity, implying that patterns of shape variation are similar across spatial scales. Steeper slopes (i.e., those larger than -1.0) correspond to data that
are globally integrated, while shallower slopes to data that are 'disintegrated (see Bookstein 2015). Isotropic data will have an expected slope of zero.

Author(s)

Dean Adams

References


Examples

data(plethodon)
Y.gpa<-gpagen(plethodon$land)  #GPA-alignment

globalIntegration(Y.gpa$coords)

gpagen

Generalized Procrustes analysis of points, curves, and surfaces

Description

A general function to perform Procrustes analysis of two- or three-dimensional landmark data that can include both fixed landmarks and sliding semilandmarks

Usage

gpagen(A, Proj = TRUE, ProcD = TRUE, PrinAxes = TRUE, ShowPlot = TRUE, curves = NULL, surfaces = NULL, ...)

Arguments

A An array (p x k x n) containing landmark coordinates for a set of specimens
Proj A logical value indicating whether or not the aligned Procrustes residuals should be projected into tangent space
ProcD A logical value indicating whether or not Procrustes distance should be used as the criterion for optimizing the positions of semilandmarks
PrinAxes A logical value indicating whether or not to align the shape data by principal axes
ShowPlot A logical value indicating whether or not a plot of Procrustes residuals should be displayed (calls plotAllSpecimens)
curves An optional matrix defining which landmarks should be treated as semilandmarks on boundary curves, and which landmarks specify the tangent directions for their sliding
gpagen

surfaces An optional vector defining which landmarks should be treated as semi-landmarks on surfaces

... Options to be passed to plotAllSpecimens

Details

The function performs a Generalized Procrustes Analysis (GPA) on two-dimensional or three-dimensional landmark coordinates. The analysis can be performed on fixed landmark points, semi-landmarks on curves, semilandmarks on surfaces, or any combination. To include semilandmarks on curves, one must specify a matrix defining which landmarks are to be treated as semilandmarks using the "curves=" option. Likewise, to include semilandmarks on surfaces, one must specify a vector listing which landmarks are to be treated as surface semilandmarks using the "surfaces=" option. The "ProcD=TRUE" option will slide the semilandmarks along their tangent directions using the Procrustes distance criterion, while "ProcD=FALSE" will slide the semilandmarks based on minimizing bending energy. The aligned Procrustes residuals can be projected into tangent space using the "Proj=TRUE" option. NOTE: Large datasets may exceed the memory limitations of R.

Generalized Procrustes Analysis (GPA: Gower 1975, Rohlf and Slice 1990) is the primary means by which shape variables are obtained from landmark data (for a general overview of geometric morphometrics see Bookstein 1991, Rohlf and Marcus 1993, Adams et al. 2004, Zelditch et al. 2012, Mitteroecker and Gunz 2009, Adams et al. 2013). GPA translates all specimens to the origin, scales them to unit-centroid size, and optimally rotates them (using a least-squares criterion) until the coordinates of corresponding points align as closely as possible. The resulting aligned Procrustes coordinates represent the shape of each specimen, and are found in a curved space related to Kendall’s shape space (Kendall 1984). Typically, these are projected into a linear tangent space yielding Kendall’s tangent space coordinates (Dryden and Mardia 1993, Rohlf 1999), which are used for subsequent multivariate analyses. Additionally, any semilandmarks on curves and are slid along their tangent directions or tangent planes during the superimposition (see Bookstein 1997; Gunz et al. 2005). Presently, two implementations are possible: 1) the locations of semilandmarks can be optimized by minimizing the bending energy between the reference and target specimen (Bookstein 1997), or by minimizing the Procrustes distance between the two (Rohlf 2010). Note that specimens are NOT automatically reflected to improve the GPA-alignment.

Value

Function returns a list with the following components:

coords A (p x k x n) array of aligned Procrustes coordinates, where p is the number of landmark points, k is the number of landmark dimensions (2 or 3), and n is the number of specimens. The third dimension of this array contains names for each specimen if specified in the original input array

Csize A vector of centroid sizes for each specimen, containing the names for each specimen if specified in the original input array

Author(s)

Dean Adams
References


Examples

```r
# Example 1: fixed points only
data(plethodon)
gpagen(plethodon$land, PrinAxes=FALSE)
gpagen(plethodon$land, pointscale=2)

# Example 2: points and semilandmarks on curves
data(hummingbirds)

# Using Procrustes Distance for sliding
gpagen(hummingbirds$land, curves=hummingbirds$curvepts)

# Using bending energy for sliding
gpagen(hummingbirds$land, curves=hummingbirds$curvepts, ProcD=FALSE)

# Example 3: points, curves and surfaces
```
gridPar

Set up parameters for grids, points, and links in plotRefToTarget

Description

Function produces a list of parameter changes within plotRefToTarget.

Usage

gridPar(pt.bg = "gray", pt.size = 1.5, link.col = "gray", link.lwd = 2,
link.lty = 1, out.col = "gray", out.cex = 0.1, tar.pt.bg = "black",
tar.pt.size = 1, tar.link.col = "black", tar.link.lwd = 2,
tar.link.lty = 1, tar.out.col = "black", tar.out.cex = 0.1,
n.col.cell = 20, grid.col = "black", grid.lwd = 1, grid.lty = 1,
txt.adj = 0.5, txt.pos = 1, txt.cex = 0.8, txt.col = "black")

Arguments

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pt.bg</td>
<td>Background color of reference configuration points (single value or vector of values)</td>
</tr>
<tr>
<td>pt.size</td>
<td>Scale factor for reference configuration points (single value or vector of values)</td>
</tr>
<tr>
<td>link.col</td>
<td>The color of links for reference configurations (single value or vector of values)</td>
</tr>
<tr>
<td>link.lwd</td>
<td>The line weight of links for reference configurations (single value or vector of values)</td>
</tr>
<tr>
<td>link.lty</td>
<td>The line type of links for reference configurations (single value or vector of values)</td>
</tr>
<tr>
<td>out.col</td>
<td>The color of outline for reference configurations (single value or vector of values)</td>
</tr>
<tr>
<td>out.cex</td>
<td>The size of plotting symbol of outline for reference configurations (single value or vector of values)</td>
</tr>
<tr>
<td>tar.pt.bg</td>
<td>Background color of target configuration points (single value or vector of values)</td>
</tr>
<tr>
<td>tar.pt.size</td>
<td>Scale factor for target configuration points (single value or vector of values)</td>
</tr>
<tr>
<td>tar.link.col</td>
<td>The color of links for target configurations (single value or vector of values)</td>
</tr>
<tr>
<td>tar.link.lwd</td>
<td>The line weight of links for target configurations (single value or vector of values)</td>
</tr>
<tr>
<td>tar.link.lty</td>
<td>The line type of links for target configurations (single value or vector of values)</td>
</tr>
<tr>
<td>tar.out.col</td>
<td>The color of outline for target configurations (single value or vector of values)</td>
</tr>
<tr>
<td>tar.out.cex</td>
<td>The size of plotting symbol of outline for target configurations (single value or vector of values)</td>
</tr>
</tbody>
</table>
n.col.cell  The number of square cells (along x axis) for grids (single numerical value)
grid.col  The color of grid lines (single value)
grid.lwd  Scale factor for the weight of grid lines (single numerical value)
grid.lty  The line type for grid lines (single numerical value, as in base R `plot`)
txt.adj  The adjustment value of the landmark label (one or two values, as in base R `text`)
txt.pos  The position of the landmark label (single numerical value, as in base R `text`)
txt.cex  The size of the landmark label text (single numerical value, as in base R `text`)
txt.col  The color of the landmark label text (single numerical value, as in base R `text`)

Details
The function allows users to vary certain plotting parameters to produce different graphical outcomes for `plotRefToTarget`. Not all parameters need to be adjusted to use this function, as the defaults above will be used.

Author(s)
Michael Collyer & Emma Sherratt

See Also
`plotRefToTarget`

Examples

data(plethodon)
Y.gpa<-gpagen(plethodon$land)  # GPA-alignment
ref<mshape(Y.gpa$coords)
plotRefToTarget(ref,Y.gpa$coords[,39])  # default settings

# Altering points and links
GP1 <- gridPar(pt.bg = "red", pt.size = 1, link.col="blue", link.lwd=2, n.col.cell=50)
plotRefToTarget(ref,Y.gpa$coords[,39], gridPars=GP1, mag=2,
links=plethodon$links, method="TPS")

# Altering point color
GP2 <- gridPar(pt.bg = "green", pt.size = 1)
plotRefToTarget(ref,Y.gpa$coords[,39], gridPars=GP2, mag=3, method="vector")

# Altering ref and target points
GP3 <- gridPar(pt.bg = "blue", pt.size = 1.5, tar.pt.bg = "orange", tar.pt.size = 1)
plotRefToTarget(ref,Y.gpa$coords[,39], gridPars=GP3, mag=3, method="points")

# Altering outline color
GP4 <- gridPar(tar.out.col = "red", tar.out.cex = 0.3)
plotRefToTarget(ref,Y.gpa$coords[,39], gridPars=GP4, mag=3,
outline=plethodon$outline, method="TPS")
Landmark data from hummingbird bills (includes sliding semilandmarks on curves)

Author(s)
Chelsea Berns and Dean Adams

References

morphol.disparity
Morphological disparity for one or more groups of specimens

Description
Function estimates morphological disparity and performs pairwise comparisons among groups.

Usage
morphol.disparity(A, groups, iter = 999)

Arguments
A A matrix (n x [p x k]) or 3D array (p x k x n) containing GPA-aligned coordinates for a set of specimens
groups A factor defining groups
iter Number of iterations for permutation test
Details

The function estimates morphological disparity and performs pairwise comparisons to identify differences between groups. The function takes as input GPA-aligned shape data [e.g., gpagen] and a grouping factor, and estimates disparity as the Procrustes variance for each group, which is the sum of the diagonal elements of the group covariance matrix (e.g., Zelditch et al. 2012). The group Procrustes variances are used as test values, and these are then statistically evaluated through permutation, where the vectors of residuals are randomized among groups. The function can be used to obtain disparity for the whole dataset by using a dummy group factor assigning all specimens to one group, in which case only Procrustes variance is returned.

Value

Function returns a list with the following components:

- **Disp.obs**: A matrix of Procrustes variance for each group
- **Prob.Dist**: A matrix of pairwise significance levels based on permutation

Author(s)

Emma Sherratt

References


Examples

data(plethodon)
Y.gpa<-gpagen(plethodon$land)  #GPA-alignment
morphol.disparity(Y.gpa$coords, groups=plethodon$site, iter = 99)
Arguments

A1  A matrix (n x [p1 x k]) or 3D array (p1 x k x n) containing GPA-aligned coordinates for the first module

A2  A matrix (n x [p2 x k]) or 3D array (p2 x k x n) containing GPA-aligned coordinates for the second module

method  Method to estimate morphological integration; see below for details

warpgrids  A logical value indicating whether deformation grids for shapes along PLS1 should be displayed (only relevant if data for A1 or A2 [or both] were input as 3D array)

iter  Number of iterations for significance testing

label  An optional vector indicating labels for each specimen that are to be displayed

verbose  A logical value indicating whether the output is basic or verbose (method="PLS" only) (see Value below)

ShowPlot  A logical value indicating whether or not the plot should be returned

Details

The function quantifies the degree of morphological integration between two modules of shape data as defined by landmark coordinates. It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with gpagen]. The function may be used to assess the degree of morphological integration between two separate structures or between two modules defined within the same landmark configuration.

Two analytical approaches are currently implemented to assess the degree of morphological integration:

1. method="PLS" (default) the function estimates the degree of morphological integration using two-block partial least squares, or PLS. When used with landmark data, this analysis is referred to as singular warps analysis (Bookstein et al. 2003). When method="PLS", the scores along the X & Y PLS axes are also returned, as is a plot of PLS scores from Block1 versus Block2 along the first set of PLS axes. Thin-plate spline deformation grids along these axes are also shown (if data were input as a 3D array). Note: deformation grids are displayed for each block of landmarks separately. If the two blocks of landmarks are derived from a single structure (i.e. a single landmark configuration), one can plot the overall deformation along PLS1 using the procedure provided in the example below.

2. method="RV" the function estimates the degree of morphological integration using the RV coefficient (Klingenberg 2009). Significance testing for both approaches is found by permuting the objects in one data matrix relative to those in the other. A histogram of coefficients obtained via resampling is presented, with the observed value designated by an arrow in the plot.

If evaluating an a priori hypothesis of modularity within a structure is of interest, one may use the average RV coefficient as implemented in the function compare.modular.partitions.
Value

Function returns the a list with the following components:

- **value**: The estimate of morphological integration: PLS.corr or RV
- **pvalue**: The significance level of the observed signal
- **xscores**: PLS scores for the first block of landmarks (method="PLS" only when verbose=TRUE)
- **yscores**: PLS scores for the second block of landmarks (method="PLS" only when verbose=TRUE)

Author(s)

Dean Adams

References


Examples

```r
data(plethodon)
Y.gpa<-gpagen(plethodon$land)  #GPA-alignment

#Morphological integration using PLS between two modules within a structure
morph.integr(Y.gpa$coords[1:5,,],Y.gpa$coords[6:12,,],method="PLS",iter=99)

#Morphological integration using RV between two modules within a structure
morph.integr(Y.gpa$coords[1:5,,],Y.gpa$coords[6:12,,],method="RV",iter=99)

#Deformation plot for case when both blocks are derived from the same landmark configuration
ref<-mshape(Y.gpa$coords)  #overall reference
plotRefToTarget(ref,Y.gpa$coords[,which.min(res$scores)],method="TPS")  #Min along PLS1
plotRefToTarget(ref,Y.gpa$coords[,which.max(res$scores)],method="TPS")  #Max along PLS1
```

Description

landmarks on mosquito wings

Author(s)

Dean Adams
**motionpaths**

*Simulated motion paths*

**Description**

Simulated motion paths

**Author(s)**

Dean Adams

**References**


---

**mshape**

*Estimate mean shape for a set of aligned specimens*

**Description**

Estimate the mean shape for a set of aligned specimens

**Usage**

`mshape(A)`

**Arguments**

`A` An array (p x k x n) containing GPA-aligned coordinates for a set of specimens

**Details**

The function estimates the average landmark coordinates for a set of aligned specimens. It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with `gpagen`]. This function is described in Claude (2008).

**Author(s)**

Julien Claude

**References**

phylo.pls

Quantify phylogenetic morphological integration between two sets of variables

Examples

data(plethodon)
Y.gpa<-gpagen(plethodon$land)  # GPA-alignment
mshape(Y.gpa$coords)  # mean (consensus) configuration

Description

Function quantifies the degree of phylogenetic morphological covariation between two sets of Procrustes-aligned coordinates using partial least squares.

Usage

phylo.pls(A1, A2, phy, warpgrids = TRUE, iter = 999, label = NULL, verbose = FALSE, ShowPlot = TRUE)

Arguments

A1 A 2D array (n x [p1 x k1]) or 3D array (p1 x k1 x n) containing landmark coordinates for the first block
A2 A 2D array (n x [p2 x k2]) or 3D array (p2 x k2 x n) containing landmark coordinates for the second block
phy A phylogenetic tree of class phylo - see read.tree in library ape
warpgrids A logical value indicating whether deformation grids for shapes along PC1 should be displayed (only relevant if data for A1 or A2 [or both] were input as 3D array)
iter Number of iterations for significance testing
label An optional vector indicating labels for each specimen that are to be displayed
verbose A logical value indicating whether the output is basic or verbose (see Value below)
ShowPlot A logical value indicating whether or not the plot should be returned

Details

The function quantifies the degree of phylogenetic morphological integration between two sets of shape data as defined by landmark coordinates. It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with gpagen].

The function estimates the degree of morphological covariation between two sets of variables while accounting for phylogeny using partial least squares (Adams and Felice 2014). The observed value is statistically assessed using permutation, where data for one block are permuted across the tips.
of the phylogeny, an estimate of the covariation between sets of variables, and compared to the observed value.

A plot of PLS scores from Block1 versus Block2 is provided for the first set of PLS axes. Thin-plate spline deformation grids along these axes are also shown (if data were input as a 3D array).

**Value**

Function returns a list with the following components:

- **PLS Correlation**
  - The estimate of phylogenetic morphological covariation
- **pvalue**
  - The significance level of the observed signal
- **Block 1 PLS Scores**
  - PLS scores for the first block of landmarks (when verbose=TRUE)
- **Block 2 PLS Scores**
  - PLS scores for the second block of landmarks (when verbose=TRUE)

**Author(s)**

Dean Adams

**References**


**Examples**

```r
data(plethspecies)
Y.gpa<-gparen(plethspecies$land)  # GPA-alignment
phylo.pls(Y.gpa$coords[1:5,,],Y.gpa$coords[6:11,,],plethspecies$phy,iter=99)
```

**Description**

Function calculates the degree of phylogenetic signal from a set of Procrustes-aligned specimens

**Usage**

```r
physignal(phy, A, iter = 999, ShowPlot = TRUE, method = c("Kmult", "SSC"))
```
Arguments

phy A phylogenetic tree of class phylo - see `read.tree` in library ape
A A matrix (n x [p x k]) or 3D array (p x k x n) containing GPA-aligned coordinates for a set of specimens
iter Number of iterations for significance testing
showPlot A logical value indicating whether or not the plot should be returned
method Method for estimating phylogenetic signal (Kmult or SSC)

Details

The function estimates the degree of phylogenetic signal present in shape data for a given phylogeny. It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with `gpagen`]. Two approaches may be used to quantify phylogenetic signal. First, a multivariate version of the K-statistic may be utilized (Kmult: Adams 2014). This value evaluates the degree of phylogenetic signal in a dataset relative to what is expected under a Brownian motion model of evolution. For geometric morphometric data, the approach is a mathematical generalization of the Kappa statistic (Blomberg et al. 2003) appropriate for highly multivariate data (see Adams 2014).

The second approach estimates phylogenetic signal as the sum of squared changes (SSC) in shape along all branches of the phylogeny (Klingenberg and Gidaszewski 2010). Significance testing is found by permuting the shape data among the tips of the phylogeny. Note that this method can be quite slow as ancestral states must be estimated for every iteration.

For both approaches a plot of the specimens in tangent space with the phylogeny superimposed is included (NOTE: if ancestral states are desired, run `plotGMPHyloMorphoSpace`).

The tree must have number of tips equal to number of taxa in the data matrix (e.g. `drop.tip`). And, tip labels of the tree MUST be exactly the same as the taxa names in the landmark data matrix (check using `match`).

This function can be used with univariate data (i.e. centroid size) if imported as matrix with row-names giving the taxa names.

Value

Function returns a list with the following components:

phy.signal The estimate of phylogenetic signal
pvalue The significance level of the observed signal

Author(s)

Dean Adams

References


Examples

data(plethspecies)
Y.gpa<-.gpagen(plethspecies$land)  #GPA-alignment

#Test for phylogenetic signal in shape
distant(plethspecies$phy,Y.gpa$coords,method="Kmult",iter=99)

#Test for phylogenetic signal in size
distant(plethspecies$phy,Y.gpa$cscore,method="Kmult",iter=99)

plethodon  

Landmark data from Plethodon salamander heads

Description

Landmark data from Plethodon salamander heads

Author(s)

Dean Adams

References


plethshapefood  

Head shape and food use data from Plethodon salamanders

Description

Head shape and food use data from Plethodon salamanders

Author(s)

Dean Adams
References


Description

Average head shape and phylogenetic relationships for several Plethodon salamander species

Author(s)

Dean Adams

References


plotAllometry

Plot allometric patterns in landmark data

Description

Function plots allometry curves for a set of specimens

Usage

plotAllometry(f1, f2 = NULL, method = c("CAC", "RegScore", "PredLine"), warpgrids = TRUE, iter = 999, label = NULL, mesh = NULL, logsz = TRUE, RRPP = FALSE, verbose = FALSE)

Arguments

f1 A formula (of the form shape ~ size); shape can be a matrix (n x [p1 x k]) or 3D array (p1 x k x n) containing GPA-aligned coordinates for the specimens
f2 An optional right hand formula for groups (e.g., ~ groups); must be a single factor
method Method for estimating allometric shape components; see below for details
The function performs a regression of shape on size, and generates a plot that describes the multivariate relationship between size and shape derived from landmark data (i.e., allometry). It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with `gpagen`]. The abscissa of the plot is size (or log of size) while the ordinate represents shape [NOTE: the function takes the input size and performs log-transformation automatically by default (logsz = TRUE), as log(Size) should be used]. Three complementary approaches can be implemented to visualize allometry:

1. If "method=CAC" (the default) the function calculates the common allometric component of the shape data, which is an estimate of the average allometric trend within groups (Mitteroecker et al. 2004). The function also calculates the residual shape component (RSC) for the data.

2. If "method=RegScore" the function calculates shape scores from the regression of shape on size, and plots these versus size (Drake and Klingenberg 2008). For a single group, these shape scores are mathematically identical to the CAC (Adams et al. 2013).

3. If "method=PredLine" the function calculates predicted values from a regression of shape on size, and plots the first principal component of the predicted values versus size as a stylized graphic of the allometric trend (Adams and Nistri 2010).

For all methods, both centroid size and allometry scores are returned. Optionally, deformation grids can be requested, which display the shape of the smallest and largest specimens relative to the average specimen (using 'warpgrids=TRUE' or 'warpgrids=FALSE'). Finally, if groups are provided, the above approaches are implemented while accounting for within-group patterns of covariation (see references for explanation). In this case, the regression is of the form: shape~size+groups (Note: to examine the interaction term use `procD.lm`). Specimens from each group are plotted using distinct colors based on the order in which the groups are found in the dataset, and using R's standard color palette: black, red, green, blue, cyan, magenta, yellow, and gray.

**Value**

Function returns an ANOVA table of statistical results for size: df, SS, MS, F, Z,Prand. If verbose=TRUE, function returns a list with the following components:

- `ProcDist.lm` An ANOVA table as above
plotAllSpecimens

allom.score  A matrix of the allometry shape scores
logSize      A matrix of log size
pred.shape   A matrix containing the predicted shapes from the regression
resid.shape  The residual shape component (RSC) of the data ("method=CAC" only)

Author(s)
Dean Adams and Michael Collyer

References
Adams, D. C., and A. Nistri. 2010. Ontogenetic convergence and evolution of foot morphology in
Drake, A. G., and C. P. Klingenberg. 2008. The pace of morphological change: Historical transfor-

Examples
data(ratland)
Y.gpa<-gpagen(ratland,PrinAxes=FALSE)    #GPA-alignment

#Using CAC for plot
plotAllometry(Y.gpa$coords ~ Y.gpa$Csize, method="CAC", iter=9)

#Using Regression Scores for plot
plotAllometry(Y.gpa$coords ~ Y.gpa$Csize, method="RegScore", iter=9)

#Using predicted allometry curve for plot
plotAllometry(Y.gpa$coords ~ Y.gpa$Csize, method="PredLine", iter=9)

plotAllSpecimens  Plot landmark coordinates for all specimens

Description
Function plots landmark coordinates for a set of specimens

Usage
plotAllSpecimens(A, mean = TRUE, links = NULL, pointscale = 1,
                meansize = 2)
Arguments

- **A**: An array (p x k x n) containing GPA-aligned coordinates for a set of specimens
- **mean**: A logical value indicating whether the mean shape should be included in the plot
- **links**: An optional matrix defining for links between landmarks
- **pointscale**: An optional value defining the size of the points for all specimens
- **meansize**: An optional value defining the size of the points representing the average specimen

Details

The function creates a plot of the landmark coordinates for all specimens. This is useful for examining patterns of shape variation after GPA. If "mean=TRUE", the mean shape will be calculated and added to the plot. Additionally, if a matrix of links is provided, the landmarks of the mean shape will be connected by lines. The link matrix is an m x 2 matrix, where m is the desired number of links. Each row of the link matrix designates the two landmarks to be connected by that link. The function will plot either two- or three-dimensional data.

Author(s)

Dean Adams

Examples

```r
data(plethodon)
Y.gpa<-gpagen(plethodon$land) #GPA-alignment

plotAllSpecimens(Y.gpa$coords, links=plethodon$links)
```

**plotGMPhyloMorphoSpace**

*Plot phylogenetic tree and specimens in tangent space*

Description

Function plots a phylogenetic tree and a set of Procrustes-aligned specimens in tangent space

Usage

```
plotGMPhyloMorphoSpace(phy, A, tip.labels = TRUE, node.labels = TRUE, ancStates = TRUE, xaxis = 1, yaxis = 2, zaxis = NULL, plot.param = list(), shadow = FALSE)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phy</td>
<td>A phylogenetic tree of class phylo - see read.tree in library ape</td>
</tr>
<tr>
<td>A</td>
<td>A matrix (n x [p x k]) or 3D array (p x k x n) containing GPA-aligned coordinates for a set of specimens</td>
</tr>
<tr>
<td>tip.labels</td>
<td>A logical value indicating whether taxa labels (tips) should be included</td>
</tr>
<tr>
<td>node.labels</td>
<td>A logical value indicating whether node labels (ancestors) should be included</td>
</tr>
<tr>
<td>ancStates</td>
<td>A logical value indicating whether ancestral state values should be returned</td>
</tr>
<tr>
<td>xaxis</td>
<td>A numeric value indicating which PC axis should be displayed as the X-axis (default = PC1)</td>
</tr>
<tr>
<td>yaxis</td>
<td>A numeric value indicating which PC axis should be displayed as the Y-axis (default = PC2)</td>
</tr>
<tr>
<td>zaxis</td>
<td>Optional, a numeric value indicating which PC axis should be displayed as the Z-axis (e.g. PC3) or if zaxis=&quot;time&quot;, internal nodes are plotted along the Z-axis relative to time</td>
</tr>
<tr>
<td>plot.param</td>
<td>A list of plotting parameters for the tips (t.bg, t.pch, t.cex), nodes (n.bg, n.pch, n.cex), branches (l.col, lwd), taxa labels (txt.cex, txt.adj, txt.col) and node labels (n.txt.cex, n.txt.adj, n.txt.col)</td>
</tr>
<tr>
<td>shadow</td>
<td>A logical value indicating whether a 2D phylomorphospace should be plotted at the base when zaxis=&quot;time&quot;</td>
</tr>
</tbody>
</table>

Details

The function creates a plot of the principal dimensions of tangent space for a set of Procrustes-aligned specimens. Default is a plot of PC axis 1 and 2. The phylogenetic tree for these specimens is superimposed in this plot revealing how shape evolves (e.g., Rohlf 2002; Klingenberg and Gidaszewski 2010). The plot also displays the ancestral states for each node of the phylogenetic tree (obtained from fastAnc), whose values can optionally be returned. If a tree with branch lengths scaled by time is used, with the option zaxis = "time", the function plots a 3D phylomorphospace, with internal nodes positioned along the Z-axis scaled to time (a.k.a. Chronophylomorphospace, Sakamoto & Ruta 2012).

Value

Function returns estimated ancestral states if ancStates=TRUE

Author(s)

Dean Adams & Emma Sherratt

References


plotOutliers

Examples

data(plethspecies)
Y.gpa<--gpagen(plethspecies$land)  #GPA-alignment

plotGMPhyloMorphoSpace(plethspecies$phy,Y.gpa$coords)
plotGMPhyloMorphoSpace(plethspecies$phy,Y.gpa$coords,
  plot.param=list(t.bg="blue",txt.col="red",n.cex=1))
plotGMPhyloMorphoSpace(plethspecies$phy,Y.gpa$coords, zaxis = "time",
  plot.param=list(n.cex=2, n.bg="blue"), shadow=TRUE)

plotOutliers  Find potential outliers

Description

Function plots all specimens ordered by distance from the mean.

Usage

plotOutliers(A, groups = NULL)

Arguments

A  An array (p x k x n) containing GPA-aligned coordinates for a set of specimens

groups  An optional factor defining groups

Details

The function creates a plot of all specimens ordered by their Procrustes distance from the mean shape. The median distance (unbroken line) and upper and lower quartiles (dashed lines) summarize the distances from the mean shape. Specimens falling above the upper quartile are plotted in red. The addresses of all specimens are returned in the order displayed in the plot for further inspection by plotRefToTarget.

If the data have strong group structure and there is reasonable belief that the whole sample mean should not be used, then a factor defining the groups can be used.

Value

Function returns the landmark addresses of all specimens ordered as in the plot. If groups are used, function returns a list structure and a plot for each level in groups.

Author(s)

Emma Sherratt
plotRefToTarget

See Also
gpagen
plotTangentSpace
plotAllSpecimens

Examples
data(plethodon)
# let's make some outliers
newland <- plethodon$land
newland[,c(1,8),2] <- newland[,c(8,1),2]
newland[,c(3,11),26] <- newland[,c(11,3),2]
Y <- gpagen(newland) # GPA
out <- plotOutliers(Y$coords) # function returns dimnames and address of all specimens ordered
plotReftoTarget(mshape(Y$coords), Y$coords[,out[1]], method="vector", label=TRUE)
plotRefToTarget(mshape(Y$coords), Y$coords[,out[2]], method="vector", label=TRUE)

plotReftoTarget

Plot shape differences between a reference and target specimen

Description
Function plots shape differences between a reference and target specimen

Usage
plotRefToTarget(M1, M2, mesh = NULL, outline = NULL, method = c("TPS", "vector", "points", "surface"), mag = 1, links = NULL, label = FALSE, gridPars = NULL, useRefPts = FALSE, ...)  

Arguments

M1
Matrix of landmark coordinates for the first (reference) specimen

M2
Matrix of landmark coordinates for the second (target) specimen

mesh
A mesh3d object for use with method="surface"

outline
An x,y curve or curves warped to the reference (2D only)

method
Method used to visualize shape difference; see below for details

mag
The desired magnification to be used when visualizing the shape difference (e.g., mag=2)

links
An optional matrix defining for links between landmarks

label
A logical value indicating whether landmark numbers will be plotted

gridPars
An optional object made by gridPar

useRefPts
An option (logical value) to use reference configuration points rather than target configuration points (when method = "TPS")

...
Additional parameters not covered by gridPar to be passed to plot, plot3d or shade3d
Details

The function generates a plot of the shape differences of a target specimen relative to a reference specimen. The option mag allows the user to indicates the degree of magnification to be used when displaying the shape difference. The function will plot either two- or three-dimensional data. For two-dimensional data and thin-plate spline deformation plots, the user may also supply boundary curves of the object, which will be deformed from the reference to the target specimen using the thin-plate spline. Such curves are often useful in describing the biological shape differences expressed in the landmark coordinates. Note that to utilize this option, a boundary curve from a representative specimen must first be warped to the reference specimen using `warpRefOutline`.

Four distinct methods for plots are available:

1. TPS a thin-plate spline deformation grid is generated. For 3D data, this method will generate thin-plate spline deformations in the x-y and x-z planes. A boundary curve will also be deformed if provided by the user.

2. vector: a plot showing the vector displacements between corresponding landmarks in the reference and target specimen is shown.

3. points a plot is displayed with the landmarks in the target (black) overlaying those of the reference (gray). Additionally, if a matrix of links is provided, the landmarks of the mean shape will be connected by lines. The link matrix is an M x 2 matrix, where M is the desired number of links. Each row of the link matrix designates the two landmarks to be connected by that link.

4. surface a mesh3d surface is warped using thin-plate spline (for 3D data only). Requires mesh3d object in option mesh, made using `warpRefMesh`.

This function combines numerous plotting functions found in Claude (2008).

Value

If using method="surface", function will return the warped mesh3d object.

Author(s)

Dean Adams, Emma Sherratt & Michael Collyer

References


See Also

`gridPar`
`define.links`
`warpRefMesh`
`warpRefOutline`
**Examples**

```r
# Two dimensional data
data(plethodon)
Y.gpa<-gpagen(plethodon$land)       # GPA-alignment
ref<-mshape(Y.gpa$coords)
plotRefToTarget(ref,Y.gpa$coords[,39])
plotRefToTarget(ref,Y.gpa$coords[,39],mag=2,outline=plethodon$outline)   #magnify by 2X
plotRefToTarget(ref,Y.gpa$coords[,39],method="vector",mag=3)
plotRefToTarget(ref,Y.gpa$coords[,39],method="points",outline=plethodon$outline)
plotRefToTarget(ref,Y.gpa$coords[,39],gridPars=gridPar(pt.bg = "green", pt.size = 1),
                 method="vector",mag=3)

# Three dimensional data
data(scallops)
Y.gpa<-gpagen(A=scallops$coorddata, curves=scallops$curvslide, surfaces=scallops$surfslide)
ref<-mshape(Y.gpa$coords)
plotRefToTarget(ref,Y.gpa$coords[,1],method="points")
scallinks <- matrix(c(1,2,2,3,3,4,4,5,5,6,6,7,7,8,8,9,9,10,10,11,11,12,13,13,
                      14,14,15,15,16,16,16,16,1), nrow=16, byrow=TRUE)
plotRefToTarget(ref,Y.gpa$coords[,1],gridPars=gridPar(tar.pt.bg = "blue", tar.link.col="blue",
                                      tar.link.lwd=2), method="points", links = scallinks)
```

---

**plotspec**

*Plot 3D specimen, fixed landmarks and surface semilandmarks.*

**Description**

A function to plot three-dimensional (3D) specimen along with its landmarks.

**Usage**

```r
plotspec(spec, digitspec, fixed = NULL, ptsize = 1, centered = FALSE, ...)
```

**Arguments**

- `spec` An object of class `shape3d/mesh3d`, or matrix of 3D vertex coordinates.
- `digitspec` Name of data matrix containing 3D fixed and/or surface sliding coordinates.
- `fixed` Numeric The number of fixed template landmarks (listed first in digitspec).
- `ptszie` Numeric Size to plot the mesh points (vertices), e.g. 0.1 for dense meshes, 3 for sparse meshes.
- `centered` Logical Whether the data matrix is in the surface mesh coordinate system (centered=FALSE) or if the data were collected after the mesh was centered (centered=TRUE)-see details.
- `...` additional parameters which will be passed to `plot3d`. 
plotTangentSpace

Details
Function to plot 3D specimens along with their digitized "fixed" landmarks and semilandmarks "surface sliders" and "curve sliders". If specimen is a 3D surface (class shape3d/mesh3d) mesh is plotted. For visualization purposes, 3D coordinate data collected using *digit.fixed* or *digitsurface* and *buildtemplate* prior to build 1.1-6 were centered by default. Therefore use this function with centered=TRUE. Data collected outside geomorph should be read using centered=FALSE. The function assumes the fixed landmarks are listed at the beginning of the coordinate matrix (digitspec).

Author(s)
Erik Otarola-Castillo & Emma Sherratt

See Also
warpRefMesh
read.ply

Examples
```r
data(scallopPLY)
ply <- scallopPLY$ply
digitdat <- scallopPLY$coords
plotspec(spec=ply,digitspec=digitdat, fixed=16, centered =TRUE)
```

Description
Function plots a set of Procrustes-aligned specimens in tangent space along their principal axes

Usage
```r
plotTangentSpace(A, axis1 = 1, axis2 = 2, warpgrids = TRUE, mesh = NULL,
                 label = NULL, groups = NULL, verbose = FALSE)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>An array (p x k x n) containing landmark coordinates for a set of aligned specimens</td>
</tr>
<tr>
<td>axis1</td>
<td>A value indicating which PC axis should be displayed as the X-axis (default = PC1)</td>
</tr>
<tr>
<td>axis2</td>
<td>A value indicating which PC axis should be displayed as the Y-axis (default = PC2)</td>
</tr>
<tr>
<td>warpgrids</td>
<td>A logical value indicating whether deformation grids for shapes along X-axis should be displayed</td>
</tr>
</tbody>
</table>
plotTangentSpace

mesh A mesh3d object to be warped to represent shape deformation along X-axis (when warpgrids=TRUE) as described in `plotRefToTarget`.

label An optional vector indicating labels for each specimen are to be displayed (or if TRUE, numerical addresses are given)

groups An optional factor vector specifying group identity for each specimen

verbose A logical value indicating whether the output is basic or verbose (see Value below)

Details

The function performs a principal components analysis of shape variation and plots two dimensions of tangent space for a set of Procrustes-aligned specimens (default is PC1 vs. PC2). The percent variation along each PC-axis is returned. Additionally (and optionally, warpgrids=T), deformation grids can be requested, which display the shape of specimens at the ends of the range of variability along PC1. If groups are provided, specimens from each group are plotted using distinct colors based on the order in which the groups are found in the dataset, and using R’s standard color palette: black, red, green, blue, cyan, magenta, yellow, and gray. NOTE: to change the colors of the groups, simply substitute a vector of the desired colors for each specimen (see example below).

Value

Function returns a table summarizing the percent variation explained by each pc axis (equivalent to summary() of `prcomp`) (default, verbose = FALSE). If verbose=TRUE, function returns a list containing the following components:

pc.summary A PC summary table as above

pc.scores The set of principal component scores for all specimens.

pc.shapes A list with four components of the shape coordinates of the extreme ends of axis1 and axis2 is returned, which can be used by `warpRefMesh`.

Author(s)

Dean Adams & Emma Sherratt

Examples

data(plethodon)
Y.gpa<-gpagen(plethodon$land)  #GPA-alignment

gp <- as.factor(paste(plethodon$species, plethodon$site)) # group must be a factor
plotTangentSpace(Y.gpa$coords, groups = gp)

## To plot residual shapes from an allometry regression (note: must add mean back in!)
plotTangentSpace(arrayspecs(resid(lm(two.d.array(Y.gpa$coords)-Y.gpa$Csize))+
predict(lm(two.d.array(Y.gpa$coords)-1)),12,2))

##To change colors of groups
col.gp <- rainbow(length(levels(gp)))
  names(col.gp) <- levels(gp)
procD.lm

procD.lm <- procD.fit(formula = formula, 
iter = iter, 
rr = rr, 
int = int, 
verbose = verbose, 
...)

Description

Function performs Procrustes ANOVA with permutation procedures to assess statistical hypotheses describing patterns of shape variation and covariation for a set of Procrustes-aligned coordinates.

Usage

procD.lm(f1, iter = 999, RRPP = FALSE, int.first = FALSE, 
verbose = FALSE, ...)

Arguments

f1 A formula for the linear model (e.g., y ~ x1 + x2)
iter Number of iterations for significance testing
RRPP A logical value indicating whether residual randomization should be used for significance testing
int.first A logical value to indicate if interactions of first main effects should precede subsequent main effects
verbose A logical value specifying whether additional output should be displayed
... Arguments passed on to procD.fit (typically associated with the lm function)

Details

The function quantifies the relative amount of shape variation attributable to one or more factors in a linear model and estimates the probability of this variation ("significance") for a null model, via distributions generated from resampling permutations. Data input is specified by a formula (e.g., y ~ X), where 'y' specifies the response variables (shape data), and 'X' contains one or more independent variables (discrete or continuous). The response matrix 'y' can be either in the form of a two-dimensional data matrix of dimension (n x [p x k]), or a 3D array (p x n x k). It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with gpagen]. The names specified for the independent (x) variables in the formula represent one or more vectors containing continuous data or factors. It is assumed that the order of the specimens in the shape matrix matches the order of values in the independent variables. Linear model fits (using the lm function) can also be input in place of a formula. Arguments for lm can also be passed on via this function.
The function `two.d.array` can be used to obtain a two-dimensional data matrix from a 3D array of landmark coordinates; however this step is no longer necessary, as procD.lm can receive 3D arrays as dependent variables.

The function performs statistical assessment of the terms in the model using Procrustes distances among specimens, rather than explained covariance matrices among variables. With this approach, the sum-of-squared Procrustes distances are used as a measure of SS (see Goodall 1991). The observed SS are evaluated through permutation. In morphometrics this approach is known as a Procrustes ANOVA (Goodall 1991), which is equivalent to distance-based anova designs (Anderson 2001). Two possible resampling procedures are provided. First, if RRPP=FALSE, the rows of the matrix of shape variables are randomized relative to the design matrix. This is analogous to a ‘full’ randomization. Second, if RRPP=TRUE, a residual randomization permutation procedure is utilized (Collyer et al. 2014). Here, residual shape values from a reduced model are obtained, and are randomized with respect to the linear model under consideration. These are then added to predicted values from the remaining effects to obtain pseudo-values from which SS are calculated. NOTE: for single-factor designs, the two approaches are identical. However, when evaluating factorial models it has been shown that RRPP attains higher statistical power and thus has greater ability to identify patterns in data should they be present (see Anderson and terBraak 2003). Effect-sizes (Z-scores) are computed as standard deviates of the SS sampling distributions generated, which might be more intuitive for P-values than F-values (see Collyer et al. 2014). In the case that multiple factor or factor-covariate interactions are used in the model formula, one can specify whether all main effects should be added to the model first, or interactions should precede subsequent main effects (i.e., $Y \sim a + b + c + a:b + \ldots$, or $Y \sim a + b + a:b + c + \ldots$, respectively.)

**Value**

Function returns an ANOVA table of statistical results for all factors: df (for each factor), SS, MS, Rsquare, F ratio, Z-score, and Prand. If verbose=TRUE, random SS are provided.

**Author(s)**

Dean Adams and Michael Collyer

**References**


**Examples**

```r
### MANOVA example for Goodall's F test (multivariate shape vs. factors)
data(plethodon)
Y.gpa<-gpagen(plethodon$land)  #GPA-alignment
```
procD.pglsl(Y.gpa$coords ~ plethodon$species*plethodon$site, iter=499)

### Regression example
data(ratland)
rat.gpa<-gpagen(ratland)        #GPA-alignment

procD.pglsl(rat.gpa$coords ~ rat.gpa$Csize, iter=499)

## using RRPP
procD.pglsl(rat.gpa$coords ~ rat.gpa$Csize, iter=49, RRPP=TRUE)

__Description__

Function performs Procrustes ANOVA in a phylogenetic framework and uses permutation procedures to assess statistical hypotheses describing patterns of shape variation and covariation for a set of Procrustes-aligned coordinates

__Usage__

```r
procD.pglsl(f1, phy, iter = 999, int.first = FALSE, RRPP = FALSE, verbose = FALSE, ...)
```

__Arguments__

- **f1**: A formula for the linear model (e.g., y~x1+x2)
- **phy**: A phylogenetic tree of class phylo - see `read.tree` in library ape
- **iter**: Number of iterations for significance testing
- **int.first**: A logical value to indicate if interactions of first main effects should precede subsequent main effects
- **RRPP**: A logical value indicating whether residual randomization should be used for significance testing
- **verbose**: A logical value specifying whether additional output should be displayed
- **...**: Arguments passed on to procD.fit (typically associated with the lm function)

__Details__

The function performs ANOVA and regression models in a phylogenetic context under a Brownian motion model of evolution, in a manner that can accommodate high-dimensional datasets. The approach is derived from the statistical equivalency between parametric methods utilizing covariance matrices and methods based on distance matrices (Adams 2014). Data input is specified by a formula (e.g., y~X), where 'y' specifies the response variables (shape data), and 'X' contains one or more independent variables (discrete or continuous). The response matrix 'y' can be either in
the form of a two-dimensional data matrix of dimension (n x [p x k]), or a 3D array (p x n x k). It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with \texttt{gpagen}]. Linear model fits (using the \texttt{lm} function) can also be input in place of a formula. Arguments for \texttt{lm} can also be passed on via this function. The user must also specify a phylogeny describing the evolutionary relationships among species (of class \texttt{phylo}). Note that the specimen labels for both x and y must match the labels on the tips of the phylogeny.

From the phylogeny, a phylogenetic transformation matrix is obtained under a Brownian motion model, and used to transform the x and y variables. Next, the Gower-centered distance matrix is obtained from predicted values from the model (y~x), from which sums-of-squares, F-ratios, and $R^2$ are estimated for each factor in the model (see Adams, 2014). Data are then permuted across the tips of the phylogeny, and estimates of statistical values are obtained for the permuted data, which are compared to the observed value to assess significance.

Two possible resampling procedures are provided. First, if RRPP=FALSE, the rows of the matrix of shape variables are randomized relative to the design matrix. This is analogous to a 'full' randomization. Second, if RRPP=TRUE, a residual randomization permutation procedure is utilized (Collyer et al. 2014). Here, residual shape values from a reduced model are obtained, and are randomized with respect to the linear model under consideration. These are then added to predicted values from the remaining effects to obtain pseudo-values from which SS are calculated. NOTE: for single-factor designs, the two approaches are identical. However, when evaluating factorial models it has been shown that RRPP attains higher statistical power and thus has greater ability to identify patterns in data should they be present (see Anderson and terBraak 2003). Effect-sizes (Z-scores) are computed as standard deviates of the sampling distributions (of F values) generated, which might be more intuitive for P-values than F-values (see Collyer et al. 2014). In the case that multiple factor or factor-covariate interactions are used in the model formula, one can specify whether all main effects should be added to the model first, or interactions should precede subsequent main effects (i.e., Y ~ a + b + c + a:b + ..., or Y ~ a + b + a:b + c + ..., respectively.)

\textbf{Value}

Function returns an ANOVA table of statistical results for all factors: df (for each factor), SS, MS, F ratio, Prand, and Rsquare.

\textbf{Author(s)}

Dean Adams and Michael Collyer

\textbf{References}


\textbf{Examples}

```r
# Example of D-PGLS for high-dimensional data
data(plethspecies)
y.gpa<-gpagen(plethspecies$land)  #GPA-alignment
procD.pglis(Y.gpa$coords ~ Y.gpa$Csize,plethspecies$phy,iter=999)
```
### Example of D-PGLS for high-dimensional data, using RRPP

```r
procD.pgl(y.gpa$coords ~ y.gpa$size,plethspecies$phy,iter=999,RRPP=TRUE)
```

---

**pupfish**  
*landmarks on pupfish*

---

**Description**

Landmark data from *Cyprindon pecosensis* body shapes, with indication of Sex and Population from which fish were sampled (Marsh or Sinkhole). These data were previously aligned with GPA. Centroid size (CS) is also provided.

**Author(s)**

Michael Collyer

**References**


---

**ratland**  
*Landmark data from dataset rat*

---

**Description**

Landmark data from dataset rat

**Author(s)**

Dean Adams

**References**

read.morphologika

Read landmark data from Morphologika file

Description

Read Morphologika file (*.txt) to obtain landmark coordinates and specimen information.

Usage

read.morphologika(file)

Arguments

file A Morphologika *.txt file containing two- or three-dimensional landmark data.

Details

This function reads a *.txt file in the Morphologika format containing two- or three-dimensional landmark coordinates. Morphologika files are text files in one of the standard formats for geometric morphometrics (see O’Higgins and Jones 1998, 2006).

If the headers "[labels]", "[labelvalues]" and "[groups]" are present, then a data matrix containing all individual specimen information is returned. If the header "[wireframe]" is present, then a matrix of the landmark addresses for the wireframe is returned (see plotRefToTarget option 'links').

Value

Function returns a (p x k x n) array of the coordinate data. If other optional headers are present in the file (e.g. "[labels]" or "[wireframe]") function returns a list containing the "coords" array, and data matrix of "labels" and or "wireframe".

Author(s)

Emma Sherratt & Erik Otarola-Castillo

References


Description

A function to read ply files, which can be used for digitizing landmark coordinates or for shape warps.

Usage

read.ply(file, ShowSpecimen = TRUE, addNormals = TRUE)

Arguments

file An ASCII ply file
ShowSpecimen logical Indicating whether or not the ply file should be displayed
addNormals logical Indicating whether or not the normals of each vertex should be calculated (using addNormals)

Details

Function reads three-dimensional surface data in the form of a single ply file (Polygon File Format; ASCII format only, from 3D scanners such as NextEngine and David scanners). Vertices of the surface may then be used to digitize three-dimensional points, and semilandmarks on curves and surfaces. The surface may also be used as a mesh for visualizing 3D deformations (warpRefMesh). The function opens the ply file and plots the mesh, with faces rendered if file contains faces, and colored if the file contains vertex color. Vertex normals allow better visualization and more accurate digitizing with digit.fixed.

Value

Function returns the following components:

mesh3d list of class mesh3d- see rgl for details

Author(s)

Dean Adams & Emma Sherratt

Examples

# If the file has no mesh color, or color is undesirable, user can assign this as follows:
# Using the example scallop PLY
data(scallopPLY)
myply <- scallopPLY$ply
mply$material <- "gray" # using color word
mply$material <- "#FCE6C9" # using RGB code
**Description**

Read *.nts file to obtain landmark coordinates for a set of specimens.

**Usage**

`readland.nts(file)`

**Arguments**

- `file` A *.nts file containing two- or three-dimensional landmark data.

**Details**

Function reads a *.nts file containing a matrix of two- or three-dimensional landmark coordinates. NTS files are text files in one of the standard formats for geometric morphometrics (see Rohlf 2012). The parameter line contains 5 or 6 elements, and must begin with a "1" to designate a rectangular matrix. The second and third values designate how many specimens (n) and how many total variables (p x k) are in the data matrix. The fourth value is a "0" if the data matrix is complete and a "1" if there are missing values. If missing values are present, the '1' is followed by the arbitrary numeric code used to represent missing values (e.g., -999). These values will be replaced with "NA" in the output array. Subsequent analyses requires a full complement of data, see `estimate.missing`. The final value of the parameter line denotes the dimensionality of the landmarks (2,3) and begins with "DIM=". If specimen and variable labels are included, these are designated placing an "L" immediately following the specimen or variable values in the parameter file. The labels then precede the data matrix.

Missing data may also be represented by designating them using 'NA'. In this case, the standard NTSYS header is used with no numeric designation for missing data (i.e. the fourth value is '0'). The positions of missing landmarks may then be estimated using `estimate.missing`.

Function is for *.nts file containing landmark coordinates for multiple specimens. Note that *.dta files in the nts format written by Landmark Editor [http://graphics.idav.ucdavis.edu/research/projects/EvoMorph](http://graphics.idav.ucdavis.edu/research/projects/EvoMorph), and *.nts files written by Stratovan Checkpoint [http://www.stratovan.com/](http://www.stratovan.com/) have incorrect header notation; every header is 1 n p-x-k 1 9999 Dim=3, rather than 1 n p-x-k 0 Dim=3, which denotes that missing data is in the file even when it is not.

**Value**

Function returns a (p x k x n) array, where p is the number of landmark points, k is the number of landmark dimensions (2 or 3), and n is the number of specimens. The third dimension of this array contains names for each specimen, which are obtained from the image names in the *.nts file.

**Author(s)**

Dean Adams & Emma Sherratt
readland.tps

References

readland.tps  Read landmark data from tps file

Description
Read *.tps file to obtain landmark coordinates

Usage

readland.tps(file, specID = c("None", "ID", "imageID"), readcurves = FALSE,
warnmsg = T)

Arguments

file  A *.tps file containing two- or three-dimensional landmark data
specID  a character specifying whether to extract the specimen ID names from the ID or IMAGE lines (default is "None").
readcurves  A logical value stating whether CURVES= field and associated coordinate data will be read as semilandmarks (TRUE) or ignored (FALSE).
warnmsg  A logical value stating whether warnings should be printed

Details
This function reads a *.tps file containing two- or three-dimensional landmark coordinates. Tps files are text files in one of the standard formats for geometric morphometrics (see Rohlf 2010). Two-dimensional landmarks coordinates are designated by the identifier "LM=", while three-dimensional data are designated by "LM3=". Landmark coordinates are multiplied by their scale factor if this is provided for all specimens. If one or more specimens are missing the scale factor, landmarks are treated in their original units.

Missing data may be present in the file. In this case, they must be designated by 'NA'. The positions of missing landmarks may then be estimated using estimate.missing.

The user may specify whether specimen names are to be extracted from the 'ID=' field or 'IMAGE=' field and included in the resulting 3D array. e.g., for 'ID=' use (file, specID = "ID") and for 'IMAGE=' use (file, specID = "imageID"). The default is specID="None".

If there are curves defined in the file (i.e., CURVES= fields), the option readcurves should be used. When readcurves = TRUE, the coordinate data for the curves will be returned as semilandmarks and will be appended to the fixed landmark data. Then the user needs to use define.sliders or define.sliders to create a matrix designating how the curve points will slide (used by 'curves=' in gpagen). When readcurves = FALSE, only the landmark data are returned.

NOTE: At present, all other information that can be contained in tps files (comments, variables, radii, etc.) is ignored.
readmulti.nts

Value

Function returns a \((p \times k \times n)\) array, where \(p\) is the number of landmark points, \(k\) is the number of landmark dimensions (2 or 3), and \(n\) is the number of specimens. The third dimension of this array contains names for each specimen, which are obtained from the image names in the *.tps file.

Author(s)

Dean Adams & Emma Sherratt

References


---

readmulti.nts  Read landmark data from multiple nts files

Description

Read a list of names for several *.nts files to obtain landmark coordinates for a set of specimens

Usage

readmulti.nts(filelist)

Arguments

filelist  A list of names for the *.nts files to be read by the function. The names in the list require quotes ("), and .nts/.NTS suffix.

Details

This function reads a list containing the names of multiple *.nts files, where each contains the landmark coordinates for a single specimen. For these files, the number of variables (columns) of the data matrix will equal the number of dimensions of the landmark data \((k=2 \text{ or } 3)\). When the function is called a dialog box is opened, from which the user may select multiple *.nts files. These are then read and concatenated into a single matrix for all specimens. The parameter line contains 5 or 6 elements, and must begin with a "1" to designate a rectangular matrix. The second and third values designate how many specimens \((n)\) and how many total variables \((p \times k)\) are in the data matrix. The fourth value is a "0" if the data matrix is complete and a "1" if there are missing values. If missing values are present, the '1' is followed by the arbitrary numeric code used to represent missing values (e.g., -999). These values will be replaced with "NA" in the output array. Subsequent analyses requires a full complement of data, see estimate.missing. Missing data may be present in the file by designating them using 'NA'. In this case, the standard NTSYS header is used with no numeric designation for missing data (i.e., the fourth value is '0'). The positions of missing landmarks may then be estimated using estimate.missing.
Value

Function returns a $(p \times k \times n)$ array, where $p$ is the number of landmark points, $k$ is the number of landmark dimensions (2 or 3), and $n$ is the number of specimens. The third dimension of this array contains names for each specimen, which are obtained from the original file names.

Author(s)

Dean Adams & Emma Sherratt

scalloply

3D scan of a scallop shell from a .ply file in mesh3d format

Description

3D scan of a scallop shell from a .ply file in mesh3d format

Author(s)

Emma Sherratt

References


scallops

Landmark data from scallop shells

Description

Landmark data from scallop shells

Author(s)

Dean Adams and Erik Otarola-Castillo

References

trajectory.analysis  *Quantify and compare shape change trajectories*

**Description**

Function estimates attributes of shape change trajectories or motion trajectories for a set of Procrustes-aligned specimens and compares them statistically

**Usage**

```
trajectory.analysis(f1, estimate.traj = TRUE, traj.pts = NULL, iter = 999,
                   pca = TRUE, verbose = FALSE, group.cols = NULL, ...)
```

**Arguments**

- `f1` A formula for the linear model (e.g., `y~x1+x2`)
- `estimate.traj` A logical value indicating whether trajectories are estimated from original data; described below
- `traj.pts` An optional value specifying the number of points in each trajectory (if `estimate.traj=FALSE`)
- `iter` Number of iterations for significance testing
- `pca` A logical indicator if a principal component analysis should be performed on data
- `verbose` A logical indicator for verbose (random) output (observed cases always first)
- `group.cols` A vector of colors to use for trajectories, in the order of the levels of the grouping variable. E.g., `c("red", "blue", "orange",...)`
- `...` Arguments passed on to procD.fit (typically associated with the lm function)

**Details**

The function quantifies phenotypic shape change trajectories from a set of specimens, and assesses variation in these parameters via permutation. A shape change trajectory is defined by a sequence of shapes in tangent space. These trajectories can be quantified for various attributes (their size, orientation, and shape), and comparisons of these attribute enable the statistical comparison of shape change trajectories (see Collyer and Adams 2013; Collyer and Adams 2007; Adams and Collyer 2007; Adams and Collyer 2009).

Data input is specified by a formula (e.g., `Y~X`), where `Y` specifies the response variables (trajectory data), and `X` contains one or more independent variables (discrete or continuous). The response matrix `Y` can be either in the form of a two-dimensional data matrix of dimension `(n x [p x k])`, or a 3D array `(p x n x k)`.. The function `two.d.array` can be used to obtain a two-dimensional data matrix from a 3D array of landmark coordinates. It is assumed that the order of the specimens `Y` matches the order of specimens in `X`. Linear model fits (using the `lm` function) can also be input in place of a formula. Arguments for `lm` can also be passed on via this function.

There are two primary modes of analysis through this function. If "estimate.traj=TRUE" the function estimates shape trajectories using the least-squares means for groups, based on a two-factor
model (e.g., Y~A+B+A:B). Under this implementation, the last factor in 'X' must be the interaction term, and the preceding two factors must be the effects of interest. Covariates may be included in 'X', and must precede the factors of interest (e.g., Y~cov+A*B). In this implementation, 'Y' contains a matrix of landmark coordinates. It is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with \texttt{gpagen}].

If "estimate.traj=FALSE" the trajectories are assembled directly from the set of shapes provided in 'Y'. With this implementation, the user must specify the number of shapes that comprise each trajectory. This approach is useful when the set of shapes forming each trajectory have been quantified directly (e.g., when motion paths are compared: see Adams and Cerney 2007). With this implementation, variation in trajectory size, shape, and orientation are evaluated for each term in 'X'.(see Adams and Cerney 2007). Currently only single-factor analyses are supportd with this method.

Once the function has performed the analysis, it generates a plot of the trajectories as visualized in the space of principal components (PC1 vs. PC2). The first point in each trajectory is displayed as white, the last point is black, and any middle points on the trajectories are in gray. The colors of trajectories follow the order in which they are found in the dataset as a default, using R's standard color palette: black, red, green3, blue, cyan, magenta, yellow, and gray. However, one can override these colors with \texttt{group.cols}.

**Value**

If "estimate.traj=TRUE", the function returns a list with the following components:

- \texttt{anova.table} Procrustes ANOVA table
- \texttt{Size$Obs.dif} A matrix of pairwise differences in trajectory size
- \texttt{Size$Z} A matrix of pairwise effect sizes for differences in trajectory size
- \texttt{Size$P} A matrix of pairwise significance levels for differences in trajectory size
- \texttt{Direction$Obs.dif} A matrix of pairwise differences in trajectory orientation
- \texttt{Direction$Z} A matrix of effect sizes for differences in trajectory orientation
- \texttt{Direction$P} A matrix of pairwise significance levels for differences in trajectory orientation
- \texttt{Shape$Obs.dif} A matrix of pairwise differences in trajectory shape (if applicable)
- \texttt{Shape$Z} A matrix of pairwise effect sizes for differences in trajectory shape (if applicable)
- \texttt{Random.values} All random values for all RRPP permutations (when \texttt{verbose=TRUE})
- \texttt{plot.points} Points for plotting (either observed or PC points)
- \texttt{trajectory.points} Trajectory points for plotting (either observed or PC points; typically means of plot points)

If "estimate.traj=FALSE", the function returns a list with the following components:

- \texttt{MANOVA.location.covariation} Procrustes ANOVA table
- \texttt{ANOVA.Size} Results of permutational-ANOVA assessing variation in trajectory size
- \texttt{ANOVA.Dir} Results of permutational-ANOVA assessing variation in trajectory orientation
ANOVA.Shape  Results of permutational-ANOVA assessing variation in trajectory shape (if applicable)
random.SS.location  Random SS from RRPP permutations (when verbose=TRUE)
random.SS.size  Random SS from RRPP permutations (when verbose=TRUE)
random.SS.dir  Random SS from RRPP permutations (when verbose=TRUE)
random.SS.shape  Random SS from RRPP permutations (when verbose=TRUE)

Author(s)
Dean Adams and Michael Collyer

References

Examples
#1: Estimate trajectories from LS means in 2-factor model
data(plethodon)
Y.gpa<-two.d.array(gpagen(plethodon$land)$coords)

trajectory.analysis(Y.gpa~plethodon$species*plethodon$site,iter=15)

# Retaining random values (first sets are always observed)
tra <- trajectory.analysis(Y.gpa~plethodon$species*plethodon$site,iter=15, verbose = TRUE)
tra$anova.table
tra$Random.values

#2: Compare motion trajectories
data(motionpaths)

#Motion paths represented by 5 time points per motion
trajectory.analysis(motionpaths$trajectories~motionpaths$groups,
estimate.traj=FALSE, traj.pts=5,iter=15)
Two-block partial least squares analysis for shape data

Description

Function performs two-block partial least squares analysis to assess the degree of association between blocks of Procrustes-aligned coordinates (or other variables).

Usage

two.b.pls(A1, A2, warpgrids = TRUE, iter = 999, verbose = FALSE, label = NULL, ShowPlot = TRUE)

Arguments

A1
A matrix (n x [p x k]) or 3D array (p x k x n) containing GPA-aligned coordinates for the first block.

A2
A matrix (n x [p x k]) or 3D array (p x k x n) containing GPA-aligned coordinates for the second block.

warpgrids
A logical value indicating whether deformation grids for shapes along PC1 should be displayed (only relevant if data for A1 or A2 [or both] were input as 3D array).

iter
Number of iterations for significance testing.

verbose
A logical value indicating whether the output is basic or verbose (see Value below).

label
An optional vector indicating labels for each specimen that are to be displayed.

ShowPlot
A logical value indicating whether or not a plot of Procrustes residuals should be displayed.

Details

The function quantifies the degree of association between two blocks of shape data as defined by landmark coordinates using partial least squares (see Rohlf and Corti 2000). If geometric morphometric data are used, it is assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with gpagen]. If other variables are used, they must be input as a 2-Dimensional matrix (rows = specimens, columns = variables).

A plot of PLS scores from Block1 versus Block2 is provided for the first set of PLS axes. Thin-plate spline deformation grids along these axes are also shown (if data were input as a 3D array).

If the specimens in A1 and A2 are differently ordered, the function will reorder A2 to match A1.
two.d.array

Value

Function returns a list with the following components:

- **value**: The estimate of association between block
- **pvalue**: The significance level of the observed association
- **Xscores**: PLS scores for the first block of landmarks (when verbose=TRUE)
- **Yscores**: PLS scores for the second block of landmarks (when verbose=TRUE)

Author(s)

Dean Adams

References


Examples

data(plethshapefood)
yNgpa=gpgen(plethshapeFood$land)  #GPA-alignment

#2B-PLS between head shape and food use data
two.b.pls(Y.gpa$coords,plethshapeFood$food,iter=99)

# To obtain PLS coefficients, x and y scores, and random values
two.b.pls(Y.gpa$coords,plethshapeFood$food,iter=99, verbose = TRUE)

two.d.array

Convert (p x k x n) data array into 2D data matrix

Description

Convert a three-dimensional array of landmark coordinates into a 2-dimensional matrix

Usage

two.d.array(A)

Arguments

- **A**: An array (p x k x n) containing landmark coordinates for a set of specimens

Details

This function converts a (p x k x n) array of landmark coordinates into a 2-dimensional matrix. The latter format of the shape data is useful for performing subsequent statistical analyses in R (e.g., PCA, MANOVA, PLS, etc.). Row labels are preserved if included in the original array.
Function returns a two-dimensional matrix of dimension \((n \times [p \times k])\), where rows represent specimens and columns represent variables.

**Author(s)**

Dean Adams and Emma Sherratt

**Examples**

```r
data(plethodon)
plethodon$land  # original data in the form of 3D array
two.d.array(plethodon$land)  # Convert to a 2D data matrix
```

**Description**

A function to take a 3D mesh and use thin-plate spline method to warp the file into the estimated mean shape for a set of aligned specimens.

**Usage**

```r
warpRefMesh(mesh, mesh.coord, ref, color = NULL, centered = FALSE)
```

**Arguments**

- `mesh`: A mesh3d object (e.g. made by `read.ply`)
- `mesh.coord`: A \(p \times k\) matrix of 3D coordinates digitized on the ply file.
- `ref`: A \(p \times k\) matrix of 3D coordinates made by `mshape`
- `color`: Color to set the ply file `material`. If the ply already has color, use NULL. For ply files without color, `color=NULL` will be plotted as grey.
- `centered`: Logical If the data in `mesh.coords` were collected from a centered mesh (see details).

**Details**

Function takes a 3D mesh (class mesh3d or shape3d, e.g. from `read.ply`) and its digitized landmark coordinates and uses the thin-plate spline method (Bookstein 1989) to warp the mesh into the shape defined by a second set of landmark coordinates, usually those of the mean shape for a set of aligned specimens. It is highly recommended that the mean shape is used as the reference for warping (see Rohlf 1998). The workflow is as follows:

1. Calculate the mean shape using `mshape`
2. Choose an actual specimen to use for the warping. The specimen used as the template for this warping is recommended as one most similar in shape to the average of the sample, but can be any reasonable specimen - do this by eye, or use \texttt{findMeanSpec}

3. Warp this specimen into the mean shape using \texttt{warpRefMesh}

4. Use this average mesh where it asks for a mesh= in the analysis functions and visualization functions

For landmark coordinates digitized with geomorph digitizing functions, centered = TRUE. This refers to the specimen being centered prior to landmark acquisition in the RGL window. For landmark data collected outside of geomorph, centered=FALSE will usually be the case. The returned mesh3d object is for use in geomorph functions where shape deformations are plotted (\texttt{plotTangentSpace}, \texttt{plotAllometry}, \texttt{bilat.symmetry}, and \texttt{plotRefToTarget}).

\textbf{Value}

Function returns a mesh3d object, which is a list of class mesh3d (see rgl for details)

\textbf{Author(s)}

Emma Sherratt

\textbf{References}


\textbf{See Also}

\texttt{findMeanSpec}

\texttt{warpRefOutline} \hspace{1cm} \textit{Creates a 2D outline warped to the mean shape.}

\textbf{Description}

A function to take an outline (defined by many points) and use thin-plate spline method to warp the outline into the estimated mean shape for a set of aligned specimens.

\textbf{Usage}

\texttt{warpRefOutline(file, coord, ref)}
warpRefOutline

Arguments

- **file**: A .txt or .csv file of the outline point coordinates, or a .TPS file with OUT-LINES= or CURVES= elements
- **coord**: A p x k matrix of 2D fixed landmark coordinates
- **ref**: A p x k matrix of 2D coordinates made by *mshape*

Details

Function takes an outline (defined by many points) with a set of fixed landmark coordinates and uses the thin-plate spline method (Bookstein 1989) to warp the outline into the shape defined by a second set of landmark coordinates, usually those of the mean shape for a set of aligned specimens. It is highly recommended that the mean shape is used as the reference for warping (see Rohlf 1998). The workflow is as follows:

1. Calculate the mean shape using *mshape*
2. Choose an actual specimen to use for the warping. The specimen used as the template for this warping is recommended as one most similar in shape to the average of the sample, but can be any reasonable specimen - do this by eye, or use *findMeanSpec*
3. Warp this specimen into the mean shape using *warpRefOutline*
4. Use this average outline where it asks for a outline= in the analysis functions and visualization functions

The returned outline object is for use in geomorph functions where shape deformations are plotted (*plotTangentSpace, plotAllometry, bilat.symmetry, and plotRefToTarget*).

Value

Function returns an outline object

Author(s)

Emma Sherratt

References


See Also

- *findMeanSpec*
**Description**

Write *tps file from obtain landmark coordinates in a 3-dimensional array

**Usage**

`writeland.tps(A, file, scale = NULL)`

**Arguments**

- `A` An array (p x k x n) containing landmark coordinates for a set of specimens
- `file` Name of the *tps file to be created
- `scale` An optional vector containing the length of the scale for each specimen

**Details**

This function writes a *tps file from a 3-dimensional array (p x k x n) of landmark coordinates.

**Author(s)**

Dean Adams
Index

*Topic IO
  read.morphologika, 62
  read.ply, 63
  readland.nts, 64
  readland.tps, 65
  readmulti.nts, 66
  writeLand.tps, 76

*Topic analysis
  advanced.procD.lm, 3
  bilat.symmetry, 7
  compare.evol.rates, 11
  compare.modular.partitions, 13
  compare.multi.evol.rates, 14
  globalIntegration, 31
  gpagen, 32
  morphol.disparity, 37
  morphol.integr, 38
  phylo.pls, 42
  physignal, 43
  plotAllometry, 46
  procD.lm, 57
  procD.pgl.s, 59
  trajectory.analysis, 68
  two.b.pls, 71

*Topic datasets
  hummingbirds, 37
  mosquito, 40
  motionpaths, 41
  plethodon, 45
  plethShapeFood, 45
  plethspecies, 46
  pupfish, 61
  ratland, 61
  scallopPLY, 67
  scallops, 67

*Topic digitizing
  buildtemplate, 9
  digit.curves, 20
  digit.fixed, 22

  digitize2d, 23
  digitsurface, 25
  editTemplate, 27

*Topic utilities
  arrayspecs, 6
  define.links, 16
  define.modules, 17
  define.sliders, 18
  estimate.missing, 28
  findMeanSpec, 29
  fixed.angle, 30
  mshape, 41
  plotOutliers, 51
  two.d.array, 72
  warpRefMesh, 73
  warpRefOutline, 74

*Topic visualization
  gridPar, 35
  plotAllometry, 46
  plotAllSpecimens, 48
  plotGMPhyloMorphoSpace, 49
  plotRefToTarget, 52
  plotspec, 54
  plotTangentSpace, 55
  warpRefMesh, 73
  warpRefOutline, 74

  addNormals, 63
  advanced.procD.lm, 3
  arrayspecs, 6

  bilat.symmetry, 7, 74, 75
  buildtemplate, 9, 19, 25–27, 55

  compare.evol.rates, 11
  compare.modular.partitions, 13, 18, 39
  compare.multi.evol.rates, 14
  complete.cases, 28

  define.links, 16, 53
define.modules, 13, 14, 17
define.sliders, 10, 18, 20, 22, 24, 25, 65
digit.curves, 20, 20
digit.fixed, 9–11, 19–21, 22, 25, 26, 55, 63
digitize2d, 19–21, 23
digitsurface, 10, 11, 22, 25, 55
drop.tip, 44
digitizeRd, 19–21, 23
digitizeSurface, 10, 11, 22, 25, 55
edit.template, 10, 27
digitizeSurface, 65, 63
editTemplate, 10, 27
estimate.missing, 28, 64, 66
fastAnc, 50
findMeanSpec, 29, 74, 75
fixed.angle, 30
geomorph (geomorph-package), 3
geomorph-package, 3
globalIntegration, 31
gpagen, 4, 10, 11, 13, 15, 19–21, 28, 29, 31, 32, 38, 39, 41, 42, 44, 47, 52, 57, 60, 65, 69, 71
ggridPar, 35, 52, 53
hummingbirds, 37
lm, 4, 57, 60, 68
match, 44
morphol.disparity, 37
morphol.integr, 38
mosquito, 40
motionpaths, 41
mshape, 17, 18, 41, 73, 75
na.omit, 28
phylo.pls, 42
physignal, 43
plethodon, 45
plethshapeFood, 45
plethspecies, 46
plot, 36, 52
plot3d, 52, 54
plotAllometry, 46, 74, 75
plotAllSpecimens, 17, 32, 33, 48, 52
plotGMPhyloMorphoSpace, 44, 49
plotOutliers, 51
plotRefToTarget, 17, 35, 36, 51, 52, 56, 62, 74, 75
plotSpec, 54
plotTangentSpace, 52, 55, 74, 75
prcomp, 56
procD.lm, 4, 5, 47, 57
procD.pglS, 59
pupfish, 61
ratland, 61
read.morphologika, 62
read.ply, 9, 11, 22, 23, 25, 26, 55, 63, 73
read.tree, 11, 15, 42, 44, 50, 59
readland.nts, 64
readland.tps, 65
readmulti.nts, 66
scalloplPLY, 67
scallops, 67
shade3d, 52
text, 36
trajectory.analysis, 68
two.b.pls, 71
two.d.array, 28, 58, 68, 72
warpRefMesh, 7, 29, 30, 47, 53, 55, 56, 63, 73, 74
warpRefOutline, 53, 74, 75
writeland.tps, 76