Package 'geomorph'

June 8, 2017

Date 2017-06-02

Type Package

Title Geometric Morphometric Analyses of 2D/3D Landmark Data

Version 3.0.4

Author Dean Adams, Michael Collyer, Antigoni Kaliontzopoulou, Emma Sherratt

Maintainer Dean Adams <dcadams@iastate.edu>

Depends rgl, ape, R(>= 3.1.0)

Imports graphics, grDevices, stats, utils, jpeg, geiger, 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)

URL http://geomorphr.github.io/geomorph/

Repository CRAN

Date/Publication 2017-06-08 04:36:07 UTC

RoxygenNote 6.0.1

NeedsCompilation no

R topics documented:

`geomorph-package` *Geometric morphometric analyses for 2D/3D data*

Description

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.

geomorph TOC

geomorph-package

Author(s)

Dean C. Adams, Michael Collyer, Antigoni Kaliontzopoulou & Emma Sherratt

---

`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, but it is better to list the model with the most terms first or use a geomorph data frame), plus indication if means or slopes are to be compared among groups, with appropriate formulae to define how they should be compared.

Usage

```
advanced.procD.lm(f1, f2, groups = NULL, slope = NULL, angle.type = c("r",
  "deg", "rad"), phy = NULL, pc.shape = FALSE, iter = 999, seed = NULL,
  print.progress = TRUE, data = NULL, ...)
```

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). This argument should be left NULL unless one wishes to perform pairwise comparisons of different group levels. Note that this argument is used in conjunction with the argument, slope. If slope is NULL, a pairwise comparison test is performed on group least squares (LS) means. If slope is not NULL, this argument will designate the group levels to compare in terms of their slopes.
slope	A formula with one - and only one - covariate (e.g., ~ x3). This argument must be used in conjunction with the groups argument. It will not make sense if the groups argument is left NULL. The groups argument defines the groups; the slope argument defines for which covariate group slopes are compared. Group slopes can differ in their magnitude and direction of shape change.
angle.type	A value specifying whether directional differences between slopes should be represented by vector correlations (r), radians (rad) or degrees (deg).
phy	A phylogenetic tree of class phylo - see read.tree in library ape (optional)
pc.shape	An argument for whether analysis should be performed on the principal component scores of shape. This is a useful option if the data are high-dimensional (many more variables than observations) but will not affect results
iter	Number of iterations for significance testing
seed	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.
data	A data frame for the function environment; see geomorph.data.frame. If variables are transformed in formulae, they should also be transformed in the geomorph data frame. (See examples.)
...	Arguments passed on to procD.fit (typically associated with the lm function, such as weights or offset).

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 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 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 `procD.lm`, this function is strictly for comparison of two nested models. (Use of `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 or pairwise statistic sampling distributions generated, which might be more intuitive for P-values than F-values (see Collyer et al. 2015). If a phylogeny is used, the ANOVA Z-score is calculated from the sampling distributions of the F value, as the total SS will vary among permutations. For ANOVA Z-scores, a log-transformation is performed first, to assure a norammly distributed sampling distribution.

Pairwise tests are only performed if formulae are provided to compute such results. The generic functions, `print`, `summary`, and `plot` all work with `advanced.procD.lm`. The generic function, `plot`, produces diagnostic plots for Procrustes residuals of the linear fit.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

An optional argument for including a phylogenetic tree of class phylo is included in this function. ANOVA performed on separate PGLS models is analagous to a likelihood ratio test between models (Adams and Collyer 2017). Pairwise tests can also be performed after PGLS estimation of coefficients but users should be aware that no formal research on the statistical properties (type I error rates and statistical power) of pairwise statistics with PGLS has yet been performed. Using PGLS and analysis of pairwise statistics, therefore, assumes some risk.

Value

Function returns an ANOVA table of statistical results for model comparison: error df (for each model), SS, MS, F ratio, Z, and Prand. A list of essentially the same components as `procD.lm` is also returned, and additionally LS means or slopes, pairwise differences comparisons of these, effect sizes, and P-values may also be returned. If a group formula is provided but slope formula is null, pairwise differences are Procrustes distances between least squares (LS) means for the defined groups. If a slope formula is provided, two sets of pairwise differences, plus effect sizes and P-values, are provided. The first is for differences in slope vector length (magnitude). The length of the slope vector corresponds to the amount of shape change per unit of covariate change. Large differences correspond to differences in the amount of shape change between groups. The second is for slope vector orientation differences. Differences in the direction of shape change (covariance of shape variables) can be summarized as a vector correlation or angle between vectors. See `summary.advanced.procD.lm` for summary options.

Author(s)

Michael Collyer

References

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

Adams, D.C. and M.L. Collyer. 2017. Multivariate comparative methods: evaluations, comparisons, and recommendations. Systematic Biology. In press.

See Also

procD.lm

Examples

```
data(plethodon)
Y.gpa<-gpagen(plethodon$land)    #GPA-alignment
gdf <- geomorph.data.frame(Y.gpa, species = plethodon$species, site = plethodon$site)

# Example of a nested model comparison (as with ANOVA with RRPP)
advanced.procD.lm(coords ~ log(Csize) + species,
~ log(Csize)*species*site, iter=499, data = gdf)

# Example of a test of a factor interaction, plus pairwise comparisons
advanced.procD.lm(coords ~ site*species, ~ site + species, groups = ~site*species,
   iter=499, data = gdf)

# Example of a test of a factor interaction, plus pairwise comparisons,
# accounting for a common allometry
advanced.procD.lm(coords ~ Csize + site*species,
~ log(Csize) + site + species,
groups = ~ site*species, slope = ~log(Csize), iter = 499, data = gdf)

# Example of a test of homogeneity of slopes, plus pairwise slopes comparisons
gdf$group <- factor(paste(gdf$species, gdf$site, sep="."))
advanced.procD.lm(coords ~ log(Csize) + group,
~ log(Csize) * group,
groups = ~ group,
slope = ~ log(Csize), angle.type = "deg", iter = 499, data = gdf)

# Example of partial pairwise comparisons, given greater model complexity.
# Plus, working with class advanced.procD.lm objects.
aov.pleth <- advanced.procD.lm(coords ~ log(Csize)*site*species,
~ log(Csize) + site*species,
groups = ~ species, slope = ~ log(Csize), angle.type = "deg", iter = 499, data = gdf)

summary(aov.pleth) # ANOVA plus pairwise tests
plot(aov.pleth) # diagnostic plots
```

```
aov.pleth$slopes # extract the slope vectors
```

`arrayspecs` *Convert landmark data matrix into array (p x k x n)*

Description

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

Usage

```
arrayspecs(A, p, k, sep = NULL)
```

Arguments

`A`	A matrix containing landmark coordinates for a set of specimens
`p`	Number of landmarks
`k`	Number of dimensions (2 or 3)
`sep`	An optional argument to attempt to separate variable names into landmark dimension and landmark number variables. For example, X.1, Y.1, Z.1, X.2, Y.2, Z.2, ..., can be separated with sep = ".", such that rows of landmark configurations are labeled 1, 2, 3, ..., and columns are labeled X, Y, Z. Note, for variables, X1, Y1, Z1, X2, Y2, Z2, ..., where no separator is evident, use sep = "". Any illogical separation argument will result in unlabeled variables. If sep = NULL (the default), unlabeled variables are forced. This is a good idea if the original matrix has landmarks out of order (as the the landmark labels might not sort as expected).

Details

This function converts a matrix of landmark coordinates into a 3D array (p x k x n), 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 3D array (p x k x n), 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

See Also

`two.d.array`

Examples

```
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)
```

---

bilat.symmetry *Analysis of bilateral symmetry*

Description

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

Usage

```
bilat.symmetry(A, ind = NULL, side = NULL, replicate = NULL,
  object.sym = FALSE, land.pairs = NULL, data = NULL, iter = 999,
  seed = NULL, RRPP = TRUE, print.progress = TRUE)
```

Arguments

A	Either A 3D array (p x k x n) containing raw landmarks (requiring GPA to be performed) or a gpagen object (if GPA has been previously performed). If one wishes to incorporate semilandmarks, GPA should be performed first using gpagen. Otherwise, bilat.symmetry can perform the initial GPA, assuming all landmarks are fixed. For "object.sym = FALSE, landmarks should be of dimension (p x k x 2n), as each specimen is represented by both left and right configurations.
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
data	A data frame for the function environment, see geomorph.data.frame. It is imperative that the variables "ind", "side", and "replicate" in the data frame match these names exactly (as shown in examples below).
iter	Number of iterations for significance testing.

seed	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
RRPP	A logical value indicating whether residual randomization should be used for significance testing.
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

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. Statistical assessment of model effects for shape variation is accomplished using permutation procedures. 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). Details on general approaches for the study of symmetry in geometric morphometrics may be found in: Mardia et al. 2000; Klingenberg et al. 2002.

As input, the function receives either A 3D array (p x k x n) containing raw landmarks (requiring GPA to be performed) or a gpagen object (if GPA has been previously performed). If one wishes to incorporate semilandmarks, GPA should be performed first using gpagen. Otherwise, bilat.symmetry can perform the initial GPA, assuming all landmarks are fixed. For "object.sym = FALSE, landmarks should be of dimension (p x k x 2n), as each specimen is represented by both left and right configurations.

Analyses of symmetry for matched pairs of objects is implemented when object.sym=FALSE. Here, a 3D array [p x k x 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 x k x 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.

Notes for geomorph 3.0:

Compared to older versions of geomorph, some results can be expected to be slightly different. Starting with geomorph 3.0, results use only type I sums of squares (SS) with either full randomization of raw shape values or RRPP (preferred with nested terms) for analysis of variance (ANOVA). Older versions used a combination of parametric and non-parametric results, as well as a combination of type I and type III SS. While analytical conclusions should be consistent (i.e., "significance" of effects is the same), these updates maintain consistency in analytical philosophy. This change will require longer computation time for large datasets, but the trade-off allows users to have more flexibility and eliminates combining disparate analytical philosophies.

Note also that significance of terms in the model are found by comparing F-values for each term to those obtained via permutation. F-ratios and df are not strictly necessary (a ratio of SS would suffice), but they are reported as is standard for anova tables. Additionally, users will notice that the df reported are based on the number of observations rather than a combination of objects * coordinates * dimensions, as is sometimes found in morphometric studies of symmetry. However, this change has no effect on hypothesis testing, as only SS vary among permutations (df, coordinates, and dimensions are constants).

The generic functions, `print`, `summary`, and `plot` all work with `bilat.symmetry`.

Value

An object of class "bilat.symmetry" returns a list of the following

`shape.anova`	An analysis of variance table for the shape data.
`size.anova`	An analysis of variance table for the shape data (when object.sym=FALSE).
`symm.shape`	The symmetric component of shape variation.
`asym.shape`	The asymmetric component of shape variation.
`DA.component`	The directional asymmetry component, found as the mean shape for each side.
`FA.component`	The fluctuating asymmetry component for each specimen, found as the specimen-specific side deviation adjusted for the mean directional asymmetry in the dataset.
`data.type`	A value indicating whether the analysis was performed as Object or Matching symmetry.
`permutations`	The number of random permutations used.
`random.shape.F`	A matrix of random F-values from the Shape analysis.
`random.size.F`	A matrix of random F-values from the Centroid Size analysis.
`perm.method`	A value indicating whether "Raw" values were shuffled or "RRPP" performed.
`call`	The matched call.

Author(s)

Dean Adams, Emma Sherratt, and Michael Collyer

References

Klingenberg, C.P. and G.S. McIntyre. 1998. Quantitative genetics of geometric shape in the mouse mandible. Evolution. 55:2342-2352.

Mardia, K.V., F.L. Bookstein, and I.J. Moreton. 2000. Statistical assessment of bilateral symmetry of shapes. Biometrika. 87:285-300.

Klingenberg, C.P., M. Barluenga, and A. Meyer. 2002. Shape analysis of symmetric structures: quantifying variation among individuals and asymmetry. Evolution. 56:1909-1920.

Examples

```
#Example of matching symmetry

data(mosquito)
gdf <- geomorph.data.frame(wingshape = mosquito$wingshape, ind=mosquito$ind, side=mosquito$side,
replicate=mosquito$replicate)
mosquito.sym <- bilat.symmetry(A = wingshape, ind = ind, side = side,
replicate = replicate, object.sym = FALSE, RRPP = TRUE, iter = 499, data = gdf)
summary(mosquito.sym)
plot(mosquito.sym, warpgrids = TRUE)
mosquito.sym$shape.anova # extract just the anova table on shape

# Previous example, performing GPA first
Y.gpa <- gpagen(mosquito$wingshape)
mosquito.sym2 <- bilat.symmetry(A = Y.gpa, ind = ind, side = side,
replicate = replicate, object.sym = FALSE, RRPP = TRUE, iter = 499, data = gdf)
summary(mosquito.sym2)
summary(mosquito.sym) # same results

#Example of object symmetry

data(scallops)
gdf <- geomorph.data.frame(shape = scallops$coorddata, ind=scallops$ind)
scallop.sym <- bilat.symmetry(A = shape, ind = ind, object.sym = TRUE,
land.pairs=scallops$land.pairs, data = gdf, RRPP = TRUE, iter = 499)
summary(scallop.sym)

# Previous example, incorporating semilandmarks (requires GPA to be performed first)

Y.gpa <- gpagen(scallops$coorddata, curves= scallops$curvslide, surfaces = scallops$surfslide)
scallop.sym <- bilat.symmetry(A = Y.gpa, ind = ind, object.sym = TRUE,
land.pairs=scallops$land.pairs, data = gdf, RRPP = TRUE, iter = 499)
summary(scallop.sym)
# NOTE one can also: plot(scallop.sym, warpgrids = TRUE, mesh = NULL)
# NOTE one can also: scallop.sym$data.type # recall the symmetry type
```

`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
ptsize	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

Gunz P, Mitteroecker P, & Bookstein FJ (2005) Semilandmarks in Three Dimensions. Modern Morphometrics in Physical Anthropology, ed Slice DE (Springer-Verlag, New York), pp 73-98.

Mitteroecker P & Gunz P (2009) Advances in Geometric Morphometrics. Evolutionary Biology 36(2):235-247.

See Also

`read.ply`

`digit.fixed`

`digitsurface`

compare.evol.rates *Comparing net rates of shape evolution on phylogenies*

Description

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

Usage

```
compare.evol.rates(A, phy, gp, iter = 999, method = c("simulation",
  "permutation"), print.progress = TRUE)
```

Arguments

A	A 3D array (p x k x n) containing GPA-aligned coordinates for all specimens, or a matrix (n x variables)
phy	A phylogenetic tree of class phylo - see read.tree in library ape
gp	A factor array designating group membership
iter	Number of iterations for significance testing
method	One of "simulation" or "permutation", to choose which approach should be used to assess significance.
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

Details

The function compares net 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 outer-product matrix of between species differences in morphospace after phylogenetic transformation (Adams 2014). From the data the net rate of shape evolution for each group in the multi-dimensional space 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). The function can be used with univariate data (i.e. centroid size) if imported as matrix with rownames giving the taxa names.

The generic functions, print, summary, and plot all work with compare.evol.rates. The generic function, plot, produces a histogram of random rate-ratios associated with the resampling procedure.

Notes for geomorph 3.0.4 and subsequent versions:

Significance testing is now accomplished in one of two ways. First, phylogenetic simulation may be used, in which tips data are obtained under Brownian motion using a common evolutionary rate pattern for all species on the phylogeny. Specifically, the common evolutionary rate matrix for all species is used, with the multi-dimensional rate used along the diagonal elements (see

Denton and Adams 2015). This procedure is more general than the original simulation procedure, and retains the desirable statistical properties of earlier methods, and under a wider array of data types. Second, significance may be accomplished via permutation, where data values at the tips are permuted relative to the (see Adams and Collyer 2017). This procedure is shown to retain all appropriate statistical properties, including rotation-invariance of significance levels (see results of Adams and Collyer 2017).

Value

An object of class "evolrate" returns a list with the following components:

`sigma.d.ratio`	The ratio of maximum to minimum net evolutionary rates.
`P.value`	The significance level of the observed ratio.
`sigma.d.gp`	The phylogenetic net evolutionary rate for each group of species on the phylogeny.
`random.sigma`	The sigma values found in random permutations of the resampling procedure.
`permutations`	The number of random permutations used.

Author(s)

Dean Adams & Emma Sherratt

References

Adams, D.C. 2014. Quantifying and comparing phylogenetic evolutionary rates for shape and other high-dimensional phenotypic data. Syst. Biol. 63:166-177.

Denton, J.S.S., and D.C. Adams. 2015. A new phylogenetic test for comparing multiple high-dimensional evolutionary rates suggests interplay of evolutionary rates and modularity in lanternfishes (Myctophiformes; Myctophidae). Evolution. 69:2425-2440.

Adams, D.C. and M.L. Collyer. 2017. Multivariate comparative methods: evaluations, comparisons, and recommendations. Systematic Biology. In press.

Examples

```
data(plethspecies)
Y.gpa<-gpagen(plethspecies$land)    #GPA-alignment
 gp.end<-factor(c(0,0,1,0,0,1,1,0,0))  #endangered species vs. rest
 names(gp.end)<-plethspecies$phy$tip

ER<-compare.evol.rates(A=Y.gpa$coords, phy=plethspecies$phy,method="simulation",gp=gp.end,iter=999)
summary(ER)
plot(ER)
```

compare.modular.partitions

Deprecated functions in geomorph

Description

The following function has been deprecated in geomorph

Usage

```
compare.modular.partitions()
```

Details

This function has been deprecated. Below shows the original function and the replacement function.

compare.modular.partitions now deprecated: use `modularity.test` instead

compare.multi.evol.rates

Comparing net rates of shape evolution among traits on phylogenies

Description

Function calculates net 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, iter = 999,
  print.progress = TRUE)
```

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
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)
iter	Number of iterations for significance testing
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

Details

The function compares net 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 net evolutionary rates found from the outer-product matrix of between species differences in morphospace after phylogenetic transformation (sensu Adams 2014). From the data the net 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.

The shape data may be input as either a 3D array (p x k x n) containing GPA-aligned coordinates for a set of species, or as a matrix (n x [p x 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 net 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.

The generic functions, print, summary, and plot all work with compare.multi.evol.rates. The generic function, plot, produces a histogram of random rate-ratios associated with the resampling procedure.

Value

An object of class "evolrate" returns a list of the following:

rates.all	The phylogenetic evolutionary rates for each trait.
rate.ratio	The ratio of maximum to minimum evolutionary rates.
pvalue	The significance level of the observed rate ratio.
pvalue.gps	Matrix of pairwise significance levels comparing each pair of rates.
call	The matched call.

Author(s)

Dean Adams

References

Adams, D.C. 2014. Quantifying and comparing phylogenetic evolutionary rates for shape and other high-dimensional phenotypic data. Syst. Biol. 63:166-177.

Denton, J.S.S., and D.C. Adams. 2015. A new phylogenetic test for comparing multiple high-dimensional evolutionary rates suggests interplay of evolutionary rates and modularity in lanternfishes (Myctophiformes; Myctophidae). Evolution. 69:2425-2440.

Examples

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

EMR<-compare.multi.evol.rates(A=Y.gpa$coords,gp=land.gp,
    Subset=TRUE, phy= plethspecies$phy,iter=999)
summary(EMR)
plot(EMR)
```

---

compare.pls *Comparisons of Effect Sizes from Partial Least Squares*

Description

Function performs an analysis to compare the effect sizes of two or more PLS effects

Usage

```
compare.pls(...)
```

Arguments

...	saved analyses of class pls

Details

The function statistically compares the effect sizes of two or more PLS analyses. Typically, this function might be used to compare levels of integration between two or more samples, each measuring morphological integration between different modules. In such cases, the PLS correlation coefficient, r, is not a good measure of integration effect, as its expected value is dependent on both the number of specimens and number of variables (Adams and Collyer 2016). This analysis calculates effect sizes as standard deviates, z, and performs two-sample z-tests, using the pooled standard error from the sampling distributions of the PLS analyses.

To use this function, simply perform `two.b.pls`, `integration.test`, or `phylo.integration` on as many samples as desired. Any number of objects of class pls can be input.

Similar versions of this function will be designed for alternative test statistics, in the future.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Value

An object of class compare.pls, returns a list of the following

`sample.z`	A vector of effect sizes for each sample.
`sample.r.sd`	A vector of standard deviations for each sampling distribution.
`pairwise.z`	A matrix of pairwise, two-sample z scores between all pairs of effect sizes.
`pairwise.p`	A matrix of corresponding P-values.

Author(s)

Michael Collyer

References

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

Examples

```
# Example of comparative morphological integration between pupfish head and body shapes

 data(pupfish) # GPA previously performed

 group <- factor(paste(pupfish$Pop, pupfish$Sex, sep = "."))
 levels(group)

 tail.LM <- c(1:3, 5:9, 18:38)
 head.LM <- (1:56)[-tail.LM]

 tail.coords <- pupfish$coords[tail.LM,,]
 head.coords <- pupfish$coords[head.LM,,]

 # Subset 3D array by group, returning a list of 3D arrays
 tail.coords.gp <- coords.subset(tail.coords, group)
 head.coords.gp <- coords.subset(head.coords, group)
```

```
 integ.tests <- Map(function(x,y) integration.test(x, y, iter=499), head.coords.gp, tail.coords.gp)
# the map function performs the integration test on each 3D array in the lists provided

 integ.tests$Marsh.F
 integ.tests$Marsh.M
 integ.tests$Sinkhole.F
 integ.tests$Sinkhole.M

 group.Z <- compare.pls(integ.tests)
 summary(group.Z)

 # Sexual dimorphism in morphological integration in one population
 # but not the other

 # can also list different PLS analyses, separately

 compare.pls(MF = integ.tests$Marsh.F, MM = integ.tests$Marsh.M)
```

`coords.subset` *Subset landmark coordinates via a factor*

Description

Subset (split) landmark coordinates via a grouping factor

Usage

```
coords.subset(A, group)
```

Arguments

`A`	A 3D array (p x k x n) containing landmark coordinates for a set of specimens
`group`	A grouping factor of length n, for splitting the array into sub-arrays

Details

This function splits a set of landmark coordinates into subsets, as described by a factor. The results is a list of separate sets of landmarks.

Author(s)

Michael Collyer

Examples

```
data(pupfish)
group <- factor(paste(pupfish$Pop, pupfish$Sex))
levels(group)
new.coords <- coords.subset(A = pupfish$coords, group = group)
names(new.coords) # see the list levels
# access any element by:
```

`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

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 links 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

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

Arguments

`spec`	A p x k matrix containing landmark coordinates of a single specimen (2D or 3D)
`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 `modularity.test` or `integration.test`.

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 colored 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. 1,1,1,2,2,3,3,3,2) to be used with `modularity.test` or `integration.test`.

Author(s)

Emma Sherratt

See Also

modularity.test and integration.test

---

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, nsliders, surfsliders = NULL, write.file = TRUE)
```

Arguments

landmarks	A matrix containing 2D or 3D landmark coordinates of landmarks and semi-landmarks, 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 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 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" (semi-landmarks) 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

Bookstein, F. J. 1997 Landmark Methods for Forms without Landmarks: Morphometrics of Group Differences in Outline Shape. Medical Image Analysis 1(3):225-243.

See Also

digitize2d, digit.fixed, gpagen, digit.curves

Examples

```
## (not run) Use interactive function in rgl window
 # data(scallops)
 # define.sliders(scallops$coorddata[,,1], nsliders=11,surfsliders = scallops$surfslide)
 # here the first specimen is used for plotting purposes only

## Examples of AUTO mode
 ## 1 curve of sliding semilandmark
 # Define sliders for scallopdata
 sliders = define.sliders(c(5:16,1))

 ## 2 curves of sliding semilandmarks
 # Define sliders for 10 landmarks, where LMs 1, 5, and 10 fixed
 # 2, 3, and 4 are along a curve between 1 and 5
 # and 6, 7, 8, and 9 are along a curve between 5 and 10.
 sliders = rbind(define.sliders(1:5), define.sliders(5:10))
```

---

digit.curves *Calculate semilandmarks along a curve*

Description

A function that "digitizes curves" by calculating equidistant two-dimensional or 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 = TRUE)
```

Arguments

start	A numeric vector of x,y,(z) coordinates for the landmark defining the start of the curve (can be simply first point on open outline: curve[1,])

curve	A matrix (p x k) 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 (Rohlf 2015). 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 Superimposition 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.

If unsure if the points defining the curve are ordered, then plot and colour them using the rainbow function, e.g. plot(curve, pch=19, cex=0.1, col=rainbow(nrow(outline))), and it should be easy to visualise.

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

Bookstein, F. J. 1997 Landmark Methods for Forms without Landmarks: Morphometrics of Group Differences in Outline Shape. Medical Image Analysis 1(3):225-243.

Rohlf, F.J., 2015. The tps series of software. Hystrix 26(1):9-12.

See Also

digit.fixed digitize2d

digit.fixed *Digitize 3D landmarks on mesh3d object*

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

```
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)
ptsize	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` *Digitize 2D landmarks on .jpg files*

Description

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

Usage

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

Arguments

`filelist`	A list of names of jpeg images to be digitized.
`nlandmarks`	Number of landmarks to be digitized.
`scale`	A 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
`MultScale`	A logical option indicating if the coordinates should be pre-multiplied by scale
`verbose`	logical. User decides whether to digitize in verbose or silent format (see details), default is verbose

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 `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. By default, 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. Optionally, one may have the coordinates pre-multiplied by scale by using the option MultScale=TRUE.

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
`ptsize`	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 an 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 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

Gunz P, Mitteroecker P, & Bookstein FJ (2005) Semilandmarks in Three Dimensions. Modern Morphometrics in Physical Anthropology, ed Slice DE (Springer-Verlag, New York), pp 73-98.

Mitteroecker P & Gunz P (2009) Advances in Geometric Morphometrics. Evolutionary Biology 36(2):235-247.

See Also

`buildtemplate`

`read.ply`

`digit.fixed`

`editTemplate` *Edit 3D template*

Description

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

Usage

```
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 an 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`).

Author(s)

Dean Adams

References

Claude, J. 2008. Morphometrics with R. Springer, New York.

Bookstein, F. L., K. Schafer, H. Prossinger, H. Seidler, M. Fieder, G. Stringer, G. W. Weber, J.-L. Arsuaga, D. E. Slice, F. J. Rohlf, W. Recheis, A. J. Mariam, and L. F. Marcus. 1999. Comparing frontal cranial profiles in archaic and modern Homo by morphometric analysis. Anat. Rec. (New Anat.) 257:217-224.

Gunz, P., P. Mitteroecker, S. Neubauer, G. W. Weber, and F. L. Bookstein. 2009. Principles for the virtual reconstruction of hominin crania. J. Hum. Evol. 57:48-62.

Examples

```
data(plethodon)
plethland<-plethodon$land
  plethland[3,,2]<-plethland[8,,2]<-NA  #create missing landmarks
  plethland[3,,5]<-plethland[8,,5]<-plethland[9,,5]<-NA
  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

```
findMeanSpec(A)
```

Arguments

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

Details

Function takes a 3D 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

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

Arguments

`A`	A 3D 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

Adams, D. C. 1999. Methods for shape analysis of landmark data from articulated structures. Evolutionary Ecology Research. 1:959-970.

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))
```

---

geomorph.data.frame *Create a data frame with shape data*

Description

A list similar to a data frame to facilitate analysis of shape data.

Usage

```
geomorph.data.frame(...)
```

Arguments

...	a list of objects to include in the data frame.

Details

This function produces a list that can be used like a data frame in other analytical functions. The purpose is similar to the function, `data.frame`, but without the constraint that data must conform to an n (observations) x p (variables) matrix. Rather, the list produced is constrained only by n. List objects can be Procrustes residuals (coordinates) arrays, matrices, variables, distance matrices, and phylogenetic trees. Results from `gpagen` can be directly imported into a geomorph.data.frame to utilize the coordinates and centroid size as variables. (See Examples)

Author(s)

Michael Collyer

Examples

```
data(plethodon)
Y.gpa <- gpagen(plethodon$land,PrinAxes=FALSE)
gdf <- geomorph.data.frame(Y.gpa)
attributes(gdf)

gdf <- geomorph.data.frame(Y.gpa, species = plethodon$species, site = plethodon$site)
attributes(gdf)

# Using geomorph.data.frame to facilitate analysis
procD.lm(coords ~ Csize + species * site, data = gdf)
```

`globalIntegration` *Quantify global integration relative to self-similarity*

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

Bookstein, F. L. 2015. Integration, disintegration, and self-similarity: Characterizing the scales of shape variation in landmark data. Evol. Biol.42(4): 395-426.

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, curves = NULL, surfaces = NULL, PrinAxes = TRUE,
  max.iter = NULL, ProcD = TRUE, Proj = TRUE, print.progress = TRUE)
```

Arguments

A	A 3D array (p x k x n) containing landmark coordinates for a set of specimens
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
surfaces	An optional vector defining which landmarks should be treated as semilandmarks on surfaces
PrinAxes	A logical value indicating whether or not to align the shape data by principal axes
max.iter	The maximum number of GPA iterations to perform before superimposition is halted. The final number of iterations could be larger than this, if curves or surface semilandmarks are involved.
ProcD	A logical value indicating whether or not Procrustes distance should be used as the criterion for optimizing the positions of semilandmarks
Proj	A logical value indicating whether or not the aligned Procrustes residuals should be projected into tangent space
print.progress	A logical value to indicate whether a progress bar should be printed to the screen.

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.

The generic functions, `print`, `summary`, and `plot` all work with `gpagen`. The generic function, `plot`, calls `plotAllSpecimens`.

Notes for geomorph 3.0:

Compared to older versions of geomorph, users might notice subtle differences in Procrustes residuals when using semilandmarks (curves or surfaces). This difference is a result of using recursive updates of the consensus configuration with the sliding algorithms (minimized bending energy or Procrustes distances). (Previous versions used a single consensus through the sliding algorithms.) Shape differences using the recursive updates of the consensus configuration should be highly correlated with shape differences using a single consensus during the sliding algorithm, but rotational "flutter" can be expected. This should have no qualitative effect on inferential analyses using Procrustes residuals.

Value

An object of class gpagen 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.
iter	The number of GPA iterations until convergence was found (or GPA halted).
points.VCV	Variance-covariance matrix among landmark coordinates.
points.var	Variances of landmark points.
consnsus	The consensus (mean) configuration.
p	Number of landmarks.
k	Number of landmark dimensions.
nsliders	Number of semilandmarks along curves.
nsurf	Number of semilandmarks as surface points.
data	Data frame with an n x (pk) matrix of Procrustes residuals and centroid size.
Q	Final convergence criterion value.
slide.method	Method used to slide semilandmarks.
call	The match call.

Author(s)

Dean Adams and Michael Collyer

References

Adams, D. C., F. J. Rohlf, and D. E. Slice. 2004. Geometric morphometrics: ten years of progress following the 'revolution'. It. J. Zool. 71:5-16.

Adams, D. C., F. J. Rohlf, and D. E. Slice. 2013. A field comes of age: Geometric morphometrics in the 21st century. Hystrix.24:7-14.

Bookstein, F. L. 1991. Morphometric tools for landmark data: Geometry and Biology. Cambridge Univ. Press, New York.

Bookstein, F. L. 1997. Landmark methods for forms without landmarks: morphometrics of group differences in outline shape. 1:225-243.

Dryden, I. L., and K. V. Mardia. 1993. Multivariate shape analysis. Sankhya 55:460-480.

Gower, J. C. 1975. Generalized Procrustes analysis. Psychometrika 40:33-51.

Gunz, P., P. Mitteroecker, and F. L. Bookstein. 2005. semilandmarks in three dimensions. Pp. 73-98 in D. E. Slice, ed. Modern morphometrics in physical anthropology. Klewer Academic/Plenum, New York.

Kendall, D. G. 1984. Shape-manifolds, Procrustean metrics and complex projective spaces. Bulletin of the London Mathematical Society 16:81-121.

Mitteroecker, P., and P. Gunz. 2009. Advances in geometric morphometrics. Evol. Biol. 36:235-247.

Rohlf, F. J., and D. E. Slice. 1990. Extensions of the Procrustes method for the optimal superimposition of landmarks. Syst. Zool. 39:40-59.

Rohlf, F. J., and L. F. Marcus. 1993. A revolution in morphometrics. Trends Ecol. Evol. 8:129-132.

Rohlf, F. J. 1999. Shape statistics: Procrustes superimpositions and tangent spaces. Journal of Classification 16:197-223.

Rohlf, F. J. 2010. tpsRelw: Relative warps analysis. Version 1.49. Department of Ecology and Evolution, State University of New York at Stony Brook, Stony Brook, NY.

Zelditch, M. L., D. L. Swiderski, H. D. Sheets, and W. L. Fink. 2012. Geometric morphometrics for biologists: a primer. 2nd edition. Elsevier/Academic Press, Amsterdam.

Examples

```
# Example 1: fixed points only
data(plethodon)
Y.gpa <- gpagen(plethodon$land,PrinAxes=FALSE)
summary(Y.gpa)
plot(Y.gpa)

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

# Using Procrustes Distance for sliding
Y.gpa <- gpagen(hummingbirds$land,curves=hummingbirds$curvepts)
summary(Y.gpa)
plot(Y.gpa)

# Using bending energy for sliding
Y.gpa <- gpagen(hummingbirds$land,curves=hummingbirds$curvepts,ProcD=FALSE)
summary(Y.gpa)
plot(Y.gpa)

# Example 3: points, curves and surfaces
data(scallops)

# Using Procrustes Distance for sliding
Y.gpa <- gpagen(A=scallops$coorddata, curves=scallops$curvslide, surfaces=scallops$surfslide)
# NOTE can summarize as: summary(Y.gpa)
# NOTE can plot as: plot(Y.gpa)
```

`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

pt.bg	Background color of reference configuration points (single value or vector of values)
pt.size	Scale factor for reference configuration points (single value or vector of values)
link.col	The color of links for reference configurations (single value or vector of values)
link.lwd	The line weight of links for reference configurations (single value or vector of values)
link.lty	The line type of links for reference configurations (single value or vector of values)
out.col	The color of outline for reference configurations (single value or vector of values)
out.cex	The size of plotting symbol of outline for reference configurations (single value or vector of values)
tar.pt.bg	Background color of target configuration points (single value or vector of values)
tar.pt.size	Scale factor for target configuration points (single value or vector of values)
tar.link.col	The color of links for target configurations (single value or vector of values)
tar.link.lwd	The line weight of links for target configurations (single value or vector of values)
tar.link.lty	The line type of links for target configurations (single value or vector of values)
tar.out.col	The color of outline for target configurations (single value or vector of values)
tar.out.cex	The size of plotting symbol of outline for target configurations (single value or vector of values)
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")

# Altering text labels
GP5 <- gridPar(txt.pos = 3, txt.col = "red")
plotRefToTarget(ref,Y.gpa$coords[,,39], gridPars=GP5, mag=3, method="vector", label=TRUE)
```

---

hummingbirds *Landmark data from hummingbird bills (includes sliding semilandmarks on curves)*

Description

Landmark data from hummingbird bills (includes sliding semilandmarks on curves)

Author(s)

Chelsea Berns and Dean Adams

References

Berns, C.M., and Adams, D.C. 2010. Bill shape and sexual shape dimorphism between two species of temperate hummingbirds: Archilochus alexandri (black-chinned hummingbirds) and Archilochus colubris (ruby-throated hummingbirds). The Auk. 127:626-635.

`integration.test` *Quantify morphological integration between modules*

Description

Function quantifies the degree of morphological integration between modules of Procrustes-aligned coordinates

Usage

```
integration.test(A, A2 = NULL, partition.gp = NULL, iter = 999,
  seed = NULL, print.progress = TRUE)
```

Arguments

`A`	A 3D array (p x k x n) containing GPA-aligned coordinates for all specimens, or a matrix (n x variables)
`A2`	An optional 3D array (p x k x n) containing GPA-aligned coordinates for all specimens, or a matrix (n x variables) for a second partition
`partition.gp`	A list of which landmarks (or variables) belong in which partition (e.g. A,A,A,B,B,B,C,C,C) (required when only 1 dataset provided)
`iter`	Number of iterations for significance testing
`seed`	An optional argument for setting the seed for random permutations of the re-sampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
`print.progress`	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

Details

The function quantifies the degree of morphological integration between modular partitions 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 or more sets of variables.

The function estimates the degree of morphological integration using a two-block partial least squares analysis (PLS). When used with landmark data, this analysis is referred to as singular warps analysis (Bookstein et al. 2003). If more than two partitions are defined, the average pairwise PLS

correlation is utilized as the test statistic. The observed test value is then compared to a distribution of values obtained by randomly permuting the individuals (rows) in one partition relative to those in the other. A significant result is found when the observed PLS correlation is large relative to this distribution, and implies that the structures are integrated with one another (see Bookstein et al. 2003). If only two partitions are specified, a plot of PLS scores along the first set of PLS axes is optionally displayed, and thin-plate spline deformation grids along these axes are also shown if data were input as a 3D array.

Input for the analysis can take one of two forms. First, one can input a single dataset (as a matrix or 3D array, along with a vector describing which variables correspond to which partitions (for the case of a 3D array, which landmarks belong to which partitions is specified). Alternatively, when evaluating the integration between two structures or partitions, two datasets may be provided.

The generic functions, `print`, `summary`, and `plot` all work with `integration.test`. The generic function, `plot`, produces a two-block.pls plot. This function calls `plot.pls`, which has two additional arguments (with defaults): label = NULL, warpgrids = TRUE. These arguments allow one to include a vector to label points and a logical statement to include warpgrids, respectively. Warpgrids can only be included for 3D arrays of Procrustes residuals. The plot is a plot of PLS scores from Block1 versus Block2 performed for the first set of PLS axes.

Similarity to `two.b.pls` and `compare.pls` :

Note that `integration.test` performed on two matrices or arrays returns the same results as `two.b.pls`. However, `two.b.pls` is limited to only two modules. It might be of interest with 3+ modules to perform integration tests between all pairwise comaprisons of modules. This can be done, test by test, and the levels of integration can be compared with `compare.pls`. Such results are different than using the average amount of integration, as performed by `integration.test` when more than two modules are input.

Using phylogenies and PGLS:

If one wishes to incorporate a phylogeny, `phylo.integration` is the function to use. This function is exactly the same as `integration.test` but allows PGLS estimation of PLS vectors. Because `integration.test` can be used on two blocks, `phylo.integration` likewise allows one to perform a phylogenetic two-block PLS analysis.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Value

Objects of class "pls" from integration.test return a list of the following:

r.pls	The estimate of morphological integration: PLS.corr. The mean of pairwise PLS correlations between partitions is used when there are more than two partitions.
r.pls.mat	The pairwise r.pls, if the number of partitions is greater than 2.
P.value	The empirically calculated P-value from the resampling procedure.
left.pls.vectors	The singular vectors of the left (x) block (for 2 modules only).
right.pls.vectors	The singular vectors of the right (y) block (for 2 modules only).
random.r	The correlation coefficients found in each random permutation of the resampling procedure.
XScores	Values of left (x) block projected onto singular vectors (for 2 modules only).
YScores	Values of right (y) block projected onto singular vectors (for 2 modules only).
svd	The singular value decomposition of the cross-covariances (for 2 modules only).
A1	Input values for the left block (for 2 modules only).
A2	Input values for the right block (for 2 modules only).
A1.matrix	Left block (matrix) found from A1 (for 2 modules only).
A2.matrix	Right block (matrix) found from A2 (for 2 modules only).
permutations	The number of random permutations used in the resampling procedure.
call	The match call.

Author(s)

Dean Adams

References

Bookstein, F. L., P. Gunz, P. Mitteroecker, H. Prossinger, K. Schaefer, and H. Seidler. 2003. Cranial integration in Homo: singular warps analysis of the midsagittal plane in ontogeny and evolution. J. Hum. Evol. 44:167-187.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

See Also

two.b.pls, modularity.test, phylo.pls, phylo.integration, and compare.pls

Examples

```
data(plethodon)
Y.gpa<-gpagen(plethodon$land)    #GPA-alignment
#landmarks on the skull and mandible assigned to partitions
land.gps<-c("A","A","A","A","A","B","B","B","B","B","B","B")
IT <- integration.test(Y.gpa$coords, partition.gp=land.gps, iter=999)
```

```
summary(IT) # Test summary
plot(IT) # PLS plot
IT$left.pls.vectors # extracting just the left (first block) singular vectors
```

`interlmkdist` *Calculate linear distances between landmarks*

Description

A function to calculate linear distances between a set of landmark coordinates (interlandmark distances)

Usage

```
interlmkdist(A, lmks)
```

Arguments

`A`	A 3D array (p x k x n) containing landmark coordinates for a set of specimens
`lmks`	A matrix or dataframe of landmark addresses for the start and end landmarks defining m linear measurements (can be either 2-x-m or m-x-2). Either the rows or the columns should have names 'start' and 'end' to define landmarks.

Details

Function takes a 3D array of landmark coordinates from a set of specimens and the addresses for the start and end landmarks defining linear measurements and then calculates the interlandmark distances. The function returns a matrix of linear distances for all specimens. If the 'lmks' matrix has row or column names defining the name of the linear measurements, the returned matrix will use these for column names (see example). If only two interlandmark distances, 'lmks' input must be m x 2.

Value

Function returns a matrix (n x m) of m linear distances for n specimens

Author(s)

Emma Sherratt & Michael Collyer

Examples

```
data(plethodon)
# Make a matrix defining three interlandmark distances
lmks <- matrix(c(8,9,6,12,4,2), ncol=2, byrow=TRUE,
dimnames = list(c("eyeW", "headL", "mouthL"),c("start", "end")))
# where 8-9 is eye width; 6-12 is head length; 4-2 is mouth length
# or alternatively
```

```
lmks <- data.frame(eyeW = c(8,9), headL = c(6,12), mouthL = c(4,2),
row.names = c("start", "end"))
A <- plethodon$land
lineardists <- interlmkdist(A, lmks)
```

---

larvalTails *Tail-shapes of larval salamanders*

Description

Landmark data from tails of larval Salamanders exposed to different treatments of herbicides.

Details

Data set includes tail landmarks (landmarks), index for identifying semilandmarks (sliders), and vectors for variables including herbicide treatment (Treatment) and Family (Family). The latter variable indicates the egg masses (clutches) sampled in the wild from which individual eggs were randomly assigned to treatment. See Levis et al. (2016) for more experimental details.

Author(s)

Michael Collyer

References

Levis, N.A, M.L. Schooler, J.R. Johnson, and M.L. Collyer. 2016. The effects of terrestrial and aquatic herbicides on larval salamander morphology and swim speed. Biological Journal of the Linnean Society. Accepted.

---

modularity.test *Evaluate the degree of modular signal in morphometric datasets*

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

```
modularity.test(A, partition.gp, iter = 999, CI = FALSE, seed = NULL,
  print.progress = TRUE)
```

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)
iter	Number of iterations for significance testing
CI	A logical argument indicating whether bootstrapping should be used for estimating confidence intervals
seed	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

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 what is expected under the null hypothesis of random assignment of variables to partitions (i.e., neither modular nor integrated structure). Input may be either a 2D matrix of phenotypic values, or a 3D array of aligned Procrustes coordinates. 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 CR coefficient (Adams 2016). If more than two modules are defined, the average pairwise CR coefficient is utilized. The CR 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 CR coefficient is small relative to this distribution (see Adams 2016). Such a result implies that there is significantly greater independence among modules than is expected under the null hypothesis of random associations of variables (neither modular nor integrated structure). This result is consistent with the identification of significant modular structure in the data. A histogram of coefficients obtained via resampling is presented, with the observed value designated by an arrow in the plot. For landmark data, the CR coefficient found from the average CR across a 90 degree rotation of the data is used as the test statistic (see Adams 2016). For all data, the CR coefficient is returned, and (optionally) its 95 for landmark data, estimation of the CI can take some time to compute.

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.

The generic functions, print, summary, and plot all work with modularity.test. The generic function, plot, produces a two-block.pls plot. This function calls plot.pls, which has two additional arguments (with defaults): label = NULL, warpgrids = TRUE. These arguments allow one to include a vector to label points and a logical statement to include warpgrids, respectively. Warpgrids can only be included for 3D arrays of Procrustes residuals. The plot is a plot of PLS scores from Block1 versus Block2 performed for the first set of PLS axes.

Value

An object of class "CR" is a list containing the following

`CR`	Covariance ratio: The estimate of the observed modular signal.
`CInterval`	The bootstrapped 95 percent confidence intervals of the CR, if CI = TRUE.
`CR.boot`	The bootstrapped CR values, if CI = TRUE
`P.value`	The empirically calculated P-value from the resampling procedure.
`CR.mat`	For more than two partitions, the pairwise CRs among partitions.
`random.CR`	The CR calculated in each of the random permutations of the resampling procedure.
`permutations`	The number of random permutations used in the resampling procdure.
`call`	The match call.

Author(s)

Dean Adams

References

Adams, D.C. 2016.Evaluating modularity in morphometric data: Challenges with the RV coefficient and a new test measure. Methods in Ecology and Evolution 7:565-572.

See Also

`two.b.pls`, `integration.test`, `phylo.modularity`, and `phylo.integration`

Examples

```
data(pupfish)
Y.gpa<-gpagen(pupfish$coords)    #GPA-alignment
 #landmarks on the body and operculum
land.gps<-rep('a',56); land.gps[39:48]<-'b'

MT <- modularity.test(Y.gpa$coords,land.gps,CI=FALSE,iter=199)
summary(MT) # Test summary
plot(MT) # Histogram of CR sampling distribution
# Result implies modularity present
```

`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(f1, groups = NULL, iter = 999, seed = NULL,
  data = NULL, print.progress = TRUE, ...)
```

Arguments

`f1`	A formula describing the linear model used. The left-hand portion of the formula should be a 3D array (p x k x n) containing GPA-aligned coordinates for a set of specimens, or a matrix (n x variables). The right-hand portion of the formula should be " ~1" to use the overall mean, or "~ x1 + x2 + x3 +...", where each x is a covariate or factor. (Interactions and nested terms also work.) Alternatively, one can use an object of class "procD.lm" or "advanced.procD.lm", which already has a formula defined. This is especially helpful for analyses performed with `procD.pgls`, as residuals from PGLS will be used for analysis (see examples).
`groups`	A formula designating groups, e.g., groups = ~ groups. If NULL, morphol.disparity will attempt to define groups based on the linear model formula, f1. If there are no groups inherently indicated in f1 and groups is NULL, a single Procrustes variance will be returned for the entire data set.
`iter`	Number of iterations for permutation test
`seed`	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
`data`	A data frame for the function environment, see `geomorph.data.frame`
`print.progress`	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.
`...`	Arguments passed on to procD.fit (typically associated with the lm function)

Details

The function estimates morphological disparity and performs pairwise comparisons to identify differences between groups. Morphological disparity is estimated as the Procrustes variance, overall or for groups, using residuals of a linear model fit. Procrustes variance is the same sum of the diagonal elements of the group covariance matrix divided by the number of observations in the group (e.g., Zelditch et al. 2012). The function takes as input a formula to describe the linear model fit, plus a

formulaic indication of groups (e.g., ~ groups). It is assumed that the formula describes shape data that have been GPA-aligned [e.g., gpagen], although the function can work with any multivariate data.

Absolute differences in Procrustes variances are test statistics that can be used to test differences in morphological disparity among groups. These differences are 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 "~ 1" as the right-hand portion of the formula, in which case only Procrustes variance is returned. Additionally, if the right-hand portion of the formula only contains (continuous) covariates, e.g., "~ Csize", Procrustes variance will be calculated for the whole data set or groups, after accounting for the linear regression described. Finally, different factors can be indicated in the formula and for groups, if one wishes to compare morphological disparities for groups comprising only a portion of or collapsing of the groups in a more complex model (see examples).

This function can be used with an object of class "procD.lm" or "advanced.procD.lm", if such analyses have already been performed. This is especially helpful for analyses performed with gpagen, This is specially useful for analyses performed with procD.pgls. In this case, residuals obtained from PGLS estimation of coefficients, rather than OLS estimation, will be used in the analysis. Thus, one can account for phylogeny when comparing morphological disparity among groups. However, one should be aware that this approach only adjusts expected values because of phylogeny and does not assert a null hypothesis of equal variances based on phylogenetic relatedness. (For example, specials means can be adjusted because of phylogenetic relatedness, but the null hypothesis of equal variances is conditioned on the estimation of means.)

Value

Objects of class "morphol.disparity" return a list with the following components (if groups are specified):

`Procrustes.var`	Observed Procrustes variances.
`PV.dist`	Observed pairwise absolute differences (distances) among group Procrustes variances.
`PV.dist.Pval`	P-values associated with pairwise differences.
`random.PV.dist`	Pairwise distance matrices produced in the resampling procedure.
`permutations`	Number of random permutations in resampling procedure.
`call`	The match call

Author(s)

Emma Sherratt and Michael Collyer

References

Zelditch, M. L., D. L. Swiderski, H. D. Sheets, and W. L. Fink. 2012. Geometric morphometrics for biologists: a primer. 2nd edition. Elsevier/Academic Press, Amsterdam.

Examples

```
data(plethodon)
Y.gpa<-gpagen(plethodon$land)    #GPA-alignment
gdf <- geomorph.data.frame(Y.gpa, species = plethodon$species, site = plethodon$site)

# Morphological disparity for entire data set
morphol.disparity(coords ~ 1, groups= NULL, data = gdf, iter=999)

# Morphological disparity for entire data set, accounting for allometry
morphol.disparity(coords ~ Csize, groups= NULL, data = gdf, iter=999)

# Morphological disparity without covariates, using overall mean
morphol.disparity(coords ~ 1, groups= ~ species*site, data = gdf, iter=999)

# Morphological disparity without covariates, using group means
morphol.disparity(coords ~ species*site, groups= ~species*site, data = gdf, iter=999)

# Morphological disparity of different groups than those described by the linear model
morphol.disparity(coords ~ Csize + species*site, groups= ~ species, data = gdf, iter=999)

# Extracting components
MD <- morphol.disparity(coords ~ Csize + species*site, groups= ~ species, data = gdf, iter=999)
MD$Procrustes.var # just the Procrustes variances


### Morphol.disparity can be used with procD.lm or advanced.procd.lm class objects

data(plethspecies)
Y.gpa<-gpagen(plethspecies$land)    #GPA-alignment
gp.end<-factor(c(0,0,1,0,0,1,1,0,0))  #endangered species vs. rest
names(gp.end)<-plethspecies$phy$tip

gdf <- geomorph.data.frame(Y.gpa, phy = plethspecies$phy, gp.end = gp.end)

pleth.ols <- procD.lm(coords ~ Csize + gp.end,
data = gdf, iter = 999) # ordinary least squares
pleth.pgls <- procD.pgls(coords ~ Csize + gp.end, phy = phy,
data = gdf, iter = 999) # phylogenetic generalized least squares

summary(pleth.ols)
summary(pleth.pgls)

morphol.disparity(f1 = pleth.ols, groups = ~ gp.end, data = gdf, iter = 999)
morphol.disparity(f1 = pleth.pgls, groups = ~ gp.end, data = gdf, iter = 999)
```

`morphol.integr` *Deprecated functions in geomorph*

Description

The following function has been deprecated in geomorph

Usage

```
morphol.integr()
```

Details

This function has been deprecated. Below shows the original function and the replacement function.

morphol.integr now deprecated: use `integration.test` instead

---

mosquito *Landmarks on mosquito wings*

Description

Landmarks on mosquito wings

Author(s)

Dean Adams

---

motionpaths *Simulated motion paths*

Description

Simulated motion paths

Author(s)

Dean Adams

References

Adams, D. C., and M. L. Collyer. 2009. A general framework for the analysis of phenotypic trajectories in evolutionary studies. Evolution 63:1143-1154.

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	Either a list (length n, ach p x k), A 3D array (p x k x n), or a matrix (pk 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

Claude, J. 2008. Morphometrics with R. Springer, New York.

Examples

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

mshape(Y.gpa$coords)   #mean (consensus) configuration
```

`nested.update` *Update procD.lm objects with nested effects*

Description

This function is used to update `procD.lm` objects for fixed effects with nested random effects (nested design)

Usage

```
nested.update(P, f1)
```

Arguments

`P`	A `procD.lm` object
`f1`	A right-hand or full formula for one factor nested within another; e.g., ~ A/B or ~B + A/B

Details

This functions serves as a helper function when linear models have nested (hierarchical) structure. It is used on `procD.lm` objects that were formerly evaluated with type I sums of squares (SS), as is typical with models with only fixed effects. Using a formula for nested effects, this function identifies the fixed and random SS in the random outcomes used to generate the `procD.lm` object, and updates the F-values, Z-scores, and P-values based on F values adjusted to be MS fixed/MS random (nested). This is accomplished by generating random values for each iteration previously used in the `procD.lm` object.

This function can be used recursively for multiple updates, when multiple nested effects are used. The function can currently only handle single factors nested within other single factors.

Function returns the same list as `procD.lm` but with new random F values and Cohen's f-squared values substituted The ANOVA table is updated in terms of F-values, Z-scores, and P-values. Z-scores are re-calculated for all effects to be consistent with the type of distribution used. If either Cohen's f-squared values or F values were originally chosen, the same distributions are used in the update; if SS values were originally chosen, the distribution is changed to Cohen's f-squared to calculate Z-scores. This change assures consistency in effect size estimation, as the effectthat is updated cannot have an effect size based on SS.

It is important that the formula is input correctly. It can be input as one of the following four styles:

~ fixed/random

~ fixed + fixed/random

The two formulae above achieve the same model terms for the expanded model: ~ fixed + fixed:random

~ random + fixed/random

~ random + fixed + fixed/random

The two formulae above achieve the same model terms for the expanded model: ~ random + fixed + random:fixed

The procD.lm object will be updated in the same way with either of the approaches. First, the F-value for the fixed term will be adjusted as MS-fixed/MS-interaction for every random permutation. Second, the P-value for the fixed effect will be estimated from this new distribution of F-values. Although the function will try to catch improper formulae and alert the user, it is possible the function will work with an improper formula. Thus, adherence to one of the formulae above is recommended for best results.

Effect sizes (Z scores) are based on either the distribution of random F values or a distribution of Cohen's f-squared values, calculated in every pemutation. An attempt will be made to preserve the effect size type used in the previous procD.lm or procD.pgls analysis. However, an analysis performed in procD.lm using effect size calculated from random SS values will be updated to use random Cohen's f-squared values for all effects, to avoid having effect sizes measured from different distributions in the same analysis.

Author(s)

Michael Collyer

References

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

See Also

procD.lm

Examples

```
data(larvalTails)
Y.gpa <- gpagen(larvalTails$landmarks)
gdf <- geomorph.data.frame(Y.gpa, Treatment = larvalTails$Treatment, Family = larvalTails$Family)

# Model with fixed and nested effects
tailANOVA <- procD.lm(coords ~ Treatment/Family, iter=99, RRPP=TRUE, data=gdf)
summary(tailANOVA)

# Update for nested effects
tailANOVA <- nested.update(tailANOVA, ~ Treatment/Family)
summary(tailANOVA)

# Model with random, fixed, and nested effects
tailANOVA <- procD.lm(coords ~ Family + Treatment/Family, iter=99, RRPP=TRUE, data=gdf)
summary(tailANOVA)

# Update for nested effects
tailANOVA <- nested.update(tailANOVA, ~ Family + Treatment/Family)
summary(tailANOVA)

# One needs to be careful using this function!
```

```
tailANOVA <- procD.lm(coords ~ Csize * Treatment/Family, iter=99, RRPP=TRUE, data=gdf)

# This will not work: tailANOVA <- nested.update(tailANOVA, ~ Treatment/Family)
# The updated terms must be included as part of the original terms

tailANOVA <- procD.lm(coords ~ Csize + Treatment/Family, iter=99, RRPP=TRUE, data=gdf)
summary(tailANOVA)

# Now the format will allow an update

tailANOVA <- nested.update(tailANOVA, ~ Treatment/Family)

summary(tailANOVA)
```

`phylo.integration` *Quantify phylogenetic morphological integration between two or more sets of variables under Brownian motion*

Description

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

Usage

```
phylo.integration(A, A2 = NULL, phy, partition.gp = NULL, iter = 999,
  seed = NULL, print.progress = TRUE)
```

Arguments

`A`	A 2D array (n x [p1 x k1]) or 3D array (p1 x k1 x n) containing landmark coordinates for the first block
`A2`	An optional 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
`partition.gp`	A list of which landmarks (or variables) belong in which partition (e.g. A,A,A,B,B,B,C,C,C) (required when only 1 dataset provided)
`iter`	Number of iterations for significance testing
`seed`	An optional argument for setting the seed for random permutations of the re-sampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
`print.progress`	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

Details

The function quantifies the degree of phylogenetic morphological integration between two or more sets of shape data as defined by landmark coordinates. The approach is based on 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 function estimates the degree of morphological covariation between two or sets of variables while accounting for phylogeny using partial least squares (Adams and Felice 2014), and under a Brownian motion model of evolution. If more than two partitions are defined, the average pairwise PLS correlation is utilized as the test statistic. The observed value is statistically assessed using permutation, where data for one partition are permuted relative to the other partitions. Note that this permutation is performed on phylogenetically- transformed data, so that the probability of phylogenetic association of A vs. B is similar to that of B vs. A: i.e., prob(A,B|phy)~prob(B,A|phy); thus, shuffling the correct exchangeable units under the null hypothesis of no integration (Adams and Collyer 2017).

Input for the analysis can take one of two forms. First, one can input a single dataset (as a matrix or 3D array, along with a vector describing which variables correspond to which partitions (for the case of a 3D array, which landmarks belong to which partitions is specified). Alternatively, when evaluating the integration between two structures or partitions, two datasets may be provided.

The generic functions, print, summary, and plot all work with phylo.integration. The generic function, plot, produces a two-block.pls plot. This function calls plot.pls, which has two additional arguments (with defaults): label = NULL, warpgrids = TRUE. These arguments allow one to include a vector to label points and a logical statement to include warpgrids, respectively. Warpgrids can only be included for 3D arrays of Procrustes residuals. The plot is a plot of PLS scores from Block1 versus Block2 performed for the first set of PLS axes.

Similarity to two.b.pls and compare.pls :

Note that phylo.integration performed on two matrices or arrays returns the same results as a phylogentic varion of two.b.pls. It might be of interest with 3+ modules to perform separate phylogenetic integration tests between all pairwise comaprisons of modules. This can be done, test by test, and the levels of integration can be compared with compare.pls. Such results are different than using the average amount of integration, as performed by phylo.integration when more than two modules are input.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Value

Objects of class "pls" from integration.test return a list of the following:

r.pls	The estimate of morphological integration: PLS.corr. The mean of pairwise PLS correlations between partitions is used when there are more than two partitions.
r.pls.mat	The pairwise r.pls, if the number of partitions is greater than 2.
P.value	The empirically calculated P-value from the resampling procedure.
left.pls.vectors	The singular vectors of the left (x) block (for 2 modules only).
right.pls.vectors	The singular vectors of the right (y) block (for 2 modules only).
random.r	The correlation coefficients found in each random permutation of the resampling procedure.
XScores	Values of left (x) block projected onto singular vectors (for 2 modules only).
YScores	Values of right (y) block projected onto singular vectors (for 2 modules only).
svd	The singular value decomposition of the cross-covariances (for 2 modules only).
A1	Input values for the left block (for 2 modules only).
A2	Input values for the right block (for 2 modules only).
A1.matrix	Left block (matrix) found from A1 (for 2 modules only).
A2.matrix	Right block (matrix) found from A2 (for 2 modules only).
permutations	The number of random permutations used in the resampling procedure.
call	The match call.

Author(s)

Dean Adams

References

Adams, D.C. and R. Felice. 2014. Assessing phylogenetic morphological integration and trait covariation in morphometric data using evolutionary covariance matrices. PLOS ONE. 9(4):e94335.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

Adams, D.C. and M.L. Collyer. 2017. Multivariate comparative methods: evaluations, comparisons, and recommendations. Systematic Biology. In press.

See Also

integration.test, modularity.test, phylo.pls, and two.b.pls

Examples

```
data(plethspecies)
Y.gpa<-gpagen(plethspecies$land)    #GPA-alignment
land.gps<-c("A","A","A","A","A","B","B","B","B","B","B")

IT<- phylo.integration(Y.gpa$coords,partition.gp=land.gps,phy=plethspecies$phy,iter=999)
summary(IT) # Test summary
plot(IT) # PLS plot
```

`phylo.modularity` *Evaluate the degree of phylogenetic modular signal in morphometric datasets*

Description

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

Usage

```
phylo.modularity(A, partition.gp, phy, CI = FALSE, iter = 999,
  seed = NULL, print.progress = TRUE)
```

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)
`phy`	A phylogenetic tree of class phylo - see `read.tree` in library ape
`CI`	A logical argument indicating whether bootstrapping should be used for estimating confidence intervals
`iter`	Number of iterations for significance testing
`seed`	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
`print.progress`	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

Details

The function quantifies the degree of phylogenetic modularity in two or more hypothesized modules of shape data as defined by landmark coordinates, under a Brownian motion model of evolution. The degree of modularity is characterized by the covariance ratio covariance ratio (CR: see Adams 2016). The phylogenetic version of the approach procedure utilizes the evolutionary covariance matrix among traits found under a Brownian motion model of evolution as the basis of the analysis. This is the same matrix used to evaluate patterns of phylogenetic morphological integration as described in Adams and Felice (2014).

Input may be either a 2D matrix of phenotypic values, or a 3D array of aligned Procrustes coordinates. 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 CR coefficient (Adams 2016). If more than two modules are defined, the average pairwise CR coefficient is utilized. The CR 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 CR coefficient is small relative to this distribution (see Adams 2016). Such a result implies that there is significantly greater independence among modules than is expected under the null hypothesis of random associations of variables (neither modular nor integrated structure). This result is consistent with the identification of significant modular structure in the data. For landmark data, the CR coefficient found from the average CR across a 90 degree rotation of the data is used as the test statistic (see Adams 2016).

Value

Objects of class "CR" from modularity.test return a list of the following:

`CR`	Covariance ratio: The estimate of the observed modular signal.
`CInterval`	The bootstrapped 95 percent confidence intervals of the CR, if CI = TRUE.
`CR.boot`	The bootstrapped CR values, if CI = TRUE (For more than two partitions, this is the mean CR of pairwise CRs.)
`P.value`	The empirically calculated P-value from the resampling procedure.
`CR.mat`	For more than two partitions, the pairwise CRs among partitions.
`random.CR`	The CR calculated in each of the random permutations of the resampling procedure.
`permutations`	The number of random permutations used in the resampling procedure.
`call`	The match call.

Author(s)

Dean Adams

References

Adams, D.C. 2016.Evaluating modularity in morphometric data: Challenges with the RV coefficient and a new test measure. Methods in Ecology and Evolution. (Accepted).

Adams, D.C. and R. Felice. 2014. Assessing phylogenetic morphological integration and trait covariation in morphometric data using evolutionary covariance matrices. PLOS ONE. 9(4):e94335.

Examples

```
data(plethspecies)
Y.gpa<-gpagen(plethspecies$land)    #GPA-alignment
land.gps<-c("A","A","A","A","A","B","B","B","B","B","B")

MT <- phylo.modularity(Y.gpa$coords, partition.gp=land.gps, phy=plethspecies$phy,
CI = FALSE, iter=999)
summary(MT) # Test summary
plot(MT) # Histogram of CR sampling distribution
```

---

phylo.pls *Deprecated functions in geomorph*

Description

The following function has been deprecated in geomorph

Usage

```
phylo.pls()
```

Details

This function has been deprecated. Below shows the original function and the replacement function.

phylo.pls now deprecated: use `phylo.integration` instead

---

physignal *Assessing phylogenetic signal in morphometric data*

Description

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

Usage

```
physignal(A, phy, iter = 999, seed = NULL, print.progress = TRUE)
```

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
phy	A phylogenetic tree of class phylo - see read.tree in library ape
iter	Number of iterations for significance testing
seed	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

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]. The degree of phylogenetic signal in data is estimated using the multivariate version of the K-statistic (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).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.

This function can also be used with univariate data (i.e. centroid size) if imported as matrix with rownames giving the taxa names. In this case, the estimate of phylogenetic signal is identical to that found using the standard kappa statistic (Blomberg et al. 2003).

The generic functions, print, summary, and plot all work with physignal. The generic function, plot, produces a histogram of random K statistics, associated with the resampling procedure.

Notes for geomorph 3.0:

Compared to older versions of geomorph, the order of input variables has changed, so that it is consistent with other functions in the program. Additionally, users should note that the function physignal no longer contains multiple methods. Only Kmult is used. Thus, for older scripts method="" should be removed from the function call.

Value

Function returns a list with the following components:

phy.signal	The estimate of phylogenetic signal
pvalue	The significance level of the observed signal
random.K	Each random K-statistic from random permutations
permutations	The number of random permutations used in the resampling procedure
call	The matched call

Author(s)

Dean Adams

References

Blomberg SP, Garland T, Ives AR. 2003. Testing for phylogenetic signal in comparative data: behavioral traits are more labile. Evolution, 57:717-745.

Adams, D.C. 2014. A generalized K statistic for estimating phylogenetic signal from shape and other high-dimensional multivariate data. Systematic Biology. 63:685-697.

Examples

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

#Test for phylogenetic signal in shape
PS.shape <- physignal(A=Y.gpa$coords,phy=plethspecies$phy,iter=999)
summary(PS.shape)
plot(PS.shape)

#Test for phylogenetic signal in size
PS.size <- physignal(A=Y.gpa$Csize,phy=plethspecies$phy,iter=999)
summary(PS.size)
plot(PS.size)
```

---

plethodon *Landmark data from Plethodon salamander heads*

Description

Landmark data from Plethodon salamander heads

Author(s)

Dean Adams

References

Adams, D. C. 2004. Character displacement via aggressive interference in Appalachian salamanders. Ecology. 85:2664-2670.

Adams, D.C. 2010. Parallel evolution of character displacement driven by competitive selection in terrestrial salamanders. BMC Evolutionary Biology. 10(72)1-10.

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

Adams, D. C., and F. J. Rohlf. 2000. Ecological character displacement in Plethodon: biomechanical differences found from a geometric morphometric study. Proceedings of the National Academy of Sciences, U.S.A. 97:4106-4111

plethspecies — *Average head shape and phylogenetic relationships for several Plethodon salamander species*

Description

Average head shape and phylogenetic relationships for several Plethodon salamander species

Author(s)

Dean Adams

References

Phylogeny pruned from: Wiens et al. (2006). Evol.

Data from: Adams and Rohlf (2000); Adams et al. (2007); Arif et al. (2007) Myers and Adams (2008)

`plot.advanced.procD.lm` *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'advanced.procD.lm'
plot(x, ...)
```

Arguments

`x`	plot object (from `advanced.procD.lm`)
`...`	other arguments passed to `plot.procD.lm`

Author(s)

Michael Collyer

`plot.bilat.symmetry` *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'bilat.symmetry'
plot(x, warpgrids = TRUE, mesh = NULL, ...)
```

Arguments

`x`	plot object (from `bilat.symmetry`)
`warpgrids`	Logical argument whether to include warpgrids
`mesh`	Option to include mesh in warpgrids plots
`...`	other arguments passed to plot

Author(s)

Michael Collyer

plot.CR *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'CR'
plot(x, ...)
```

Arguments

x	plot object (from phylo.modularity)
...	other arguments passed to plot

Author(s)

Michael Collyer

plot.CR.phylo *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'CR.phylo'
plot(x, ...)
```

Arguments

x	plot object (from phylo.modularity)
...	other arguments passed to plot

Author(s)

Dean Adams

plot.evolrate *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'evolrate'
plot(x, ...)
```

Arguments

x	plot object
...	other arguments passed to plot

Author(s)

Michael Collyer

plot.gpagen *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'gpagen'
plot(x, ...)
```

Arguments

x	plot object (from gpagen)
...	other arguments passed to plotAllSpecimens

Author(s)

Michael Collyer

plot.physignal *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'physignal'
plot(x, ...)
```

Arguments

x	plot object (from physignal)
...	other arguments passed to plot

Author(s)

Michael Collyer

plot.pls *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'pls'
plot(x, label = NULL, warpgrids = TRUE, shapes = FALSE, ...)
```

Arguments

x	plot object (from phylo.integration or two.b.pls)
label	Optional vector to label points
warpgrids	Logical argument whether to include warpgrids
shapes	Logical argument whether to return the the shape coordinates of the extreme ends of axis1 and axis2
...	other arguments passed to plot

Value

If shapes = TRUE, function returns a list containing the shape coordinates of the extreme ends of axis1 and axis2 if 3D arrays were originally provided for each

Author(s)

Michael Collyer

`plot.procD.allometry` *Plot Function for geomorph*

Description

The following are brief descriptions of the different plotting methods, with references.

- 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.
- 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).
- 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).

Usage

```
## S3 method for class 'procD.allometry'
plot(x, method = c("CAC", "RegScore", "PredLine"),
  warpgrids = TRUE, label = NULL, gp.label = FALSE, pt.col = NULL,
  mesh = NULL, shapes = FALSE, ...)
```

Arguments

`x`	plot object (from `procD.allometry`)
`method`	Method for estimating allometric shape components
`warpgrids`	A logical value indicating whether deformation grids for small and large shapes should be displayed (note: if groups are provided no TPS grids are shown)
`label`	An optional vector indicating labels for each specimen that are to be displayed
`gp.label`	A logical value indicating labels for each group to be displayed (if group was originally included); "PredLine" only
`pt.col`	An optional vector of colours to use for points (as in points(bg=))
`mesh`	A mesh3d object to be warped to represent shape deformation of the minimum and maximum size if warpgrids=TRUE (see `warpRefMesh`).

shapes	Logical argument whether to return the the shape coordinates shape coordinates of the small and large shapes
...	other arguments passed to plot

Value

If shapes = TRUE, function returns a list containing the shape coordinates of the small and large shapes

Author(s)

Michael Collyer

References

Adams, D.C., F.J. Rohlf, and D.E. Slice. 2013. A field comes of age: geometric morphometrics in the 21st century. Hystrix. 24:7-14.

Adams, D. C., and A. Nistri. 2010. Ontogenetic convergence and evolution of foot morphology in European cave salamanders (Family: Plethodontidae). BMC Evol. Biol. 10:1-10.

Drake, A. G., and C. P. Klingenberg. 2008. The pace of morphological change: Historical transformation of skull shape in St Bernard dogs. Proc. R. Soc. B. 275:71-76.

Mitteroecker, P., P. Gunz, M. Bernhard, K. Schaefer, and F. L. Bookstein. 2004. Comparison of cranial ontogenetic trajectories among great apes and humans. J. Hum. Evol. 46:679-698.

---

plot.procD.lm *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'procD.lm'
plot(x, type = c("diagnostics", "regression", "PC"),
  outliers = FALSE, predictor = NULL, reg.type = c("CRC", "PredLine",
  "RegScore"), ...)
```

Arguments

x	plot object (from procD.lm)
type	Indicates which type of plot, choosing among diagnostics, regression, or principal component plots. Diagnostic plots are similar to lm diagnostic plots, but for multivariate data. Regression plots plot multivariate dispersion in some fashion against predictor values. PC plots project data onto the eigenvectors of the coavriance matrix for fitted values.

outliers	Logical argument to include outliers plot, if diagnostics are performed
predictor	An optional vector if "regression" plot type is chosen, and is a variable likely used in procD.lm. This vector is a vector of covariate values equal to the number of observations.
reg.type	If "regression" is chosen for plot type, this argument indicates whether a common regression component (CRC) plot, prediction line (Predline) plot, or regression score (RegScore) plotting is performed. These plots are the same as those available from procD.allometry without the constraint that the predictor is size.
...	other arguments passed to plot (helpful to employ different colors or symbols for different groups). See plot.default and par

Author(s)

Michael Collyer

`plot.trajectory.analysis` *Plot Function for geomorph*

Description

Plot Function for geomorph

Usage

```
## S3 method for class 'trajectory.analysis'
plot(x, group.cols = NULL,
  pt.seq.pattern = c("white", "gray", "black"), pt.scale = 1, ...)
```

Arguments

x	plot object (from trajectory.analysis)
group.cols	An optional vector of colors for group levels
pt.seq.pattern	The sequence of colors for starting, middle, and end points of trajectories, respectively. E.g., c("green", "gray", "red") for gray points but initial points with green color and end points with red color.
pt.scale	An optional value to magnify or reduce points (1 = no change)
...	other arguments passed to plot

Author(s)

Michael Collyer

plotAllometry *Deprecated functions in geomorph*

Description

The following function has been deprecated in geomorph

Usage

```
plotAllometry()
```

Details

This function has been deprecated. Below shows the original function and the replacement function.

plotAllometry now deprecated: use `procD.allometry` instead

plotAllSpecimens *Plot landmark coordinates for all specimens*

Description

Function plots landmark coordinates for a set of specimens

Usage

```
plotAllSpecimens(A, mean = TRUE, links = NULL, label = FALSE,
  plot.param = list())
```

Arguments

A	A 3D 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 (only if mean=TRUE)
label	A logical value indicating whether landmark numbers will be plotted (only if mean=TRUE)
plot.param	A list of plotting parameters for the points (pt.bg, pt.cex), mean (mean.bg, mean.cex), links (link.col, link.lwd, link.lty) and landmark labels (txt.cex, txt.adj, txt.pos, txt.col)

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 (e.g. see `define.links`).

Author(s)

Dean Adams

Examples

```
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

`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
`tip.labels`	A logical value indicating whether taxa labels (tips) should be included
`node.labels`	A logical value indicating whether node labels (ancestors) should be included
`ancStates`	Either a logical value indicating whether ancestral state values should be returned, or a matrix of ancestral states (i.e. calculated with `fastAnc` or `ace`)
`xaxis`	A numeric value indicating which PC axis should be displayed as the X-axis (default = PC1)
`yaxis`	A numeric value indicating which PC axis should be displayed as the Y-axis (default = PC2)

zaxis	Optional, a numeric value indicating which PC axis should be displayed as the Z-axis (e.g. PC3) or if zaxis="time", internal nodes are plotted along the Z-axis relative to time
plot.param	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)
shadow	A logical value indicating whether a 2D phylomorphospace should be plotted at the base when zaxis="time"

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 (analogous to from fastAnc from phytools), 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

Klingenberg, C. P., and N. A. Gidaszewski. 2010. Testing and quantifying phylogenetic signals and homoplasy in morphometric data. Syst. Biol. 59:245-261.

Rohlf, F. J. 2002. Geometric morphometrics and phylogeny. Pp.175-193 in N. Macleod, and P. Forey, eds. Morphology, shape, and phylogeny. Taylor & Francis, London.

Sakamoto, M. and Ruta, M. 2012. Convergence and Divergence in the Evolution of Cat Skulls: Temporal and Spatial Patterns of Morphological Diversity. PLoSONE 7(7): e39752.

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))
#NOTE: 3D plot also available: 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`	A 3D 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

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,
  axes = 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
axes	A logical value indicating whether the box and axes should be plotted (points and vector only)
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

Claude, J. 2008. Morphometrics with R. Springer, New York.

See Also

gridPar

define.links

warpRefMesh

warpRefOutline

Examples

```
# 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$coords[,,1],method="points")
# scallinks <- matrix(c(1,rep(2:16, each=2),1), nrow=16, byrow=TRUE)
# plotRefToTarget(ref,Y.gpa$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

```
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)
`ptsize`	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`.

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

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

---

plotTangentSpace *Plot specimens in tangent space*

Description

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

Usage

```
plotTangentSpace(A, axis1 = 1, axis2 = 2, warpgrids = TRUE, mesh = NULL,
  label = NULL, groups = NULL, legend = FALSE, ...)
```

Arguments

A	A 3D array (p x k x n) containing landmark coordinates for a set of aligned specimens
axis1	A value indicating which PC axis should be displayed as the X-axis (default = PC1)
axis2	A value indicating which PC axis should be displayed as the Y-axis (default = PC2)

warpgrids	A logical value indicating whether deformation grids for shapes along X-axis should be displayed
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 (see example)
legend	A logical value for whether to add a legend to the plot (only when groups are assigned).
...	Arguments passed on to prcomp. By default, plotTangentSpace will attempt to remove redundant axes (eigen values effectively 0). To override this, adjust the argument, tol, from prcomp.

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).

NOTE: previous versions of plotTangentSpace had option 'verbose' to return the PC scores and PC shapes. From version 3.0.2 this is automatic when assigned to an object.

Value

If user assigns function to object, returned is a list of the following components:

pc.summary	A table summarizing the percent variation explained by each pc axis, equivalent to summary of prcomp.
pc.scores	The set of principal component scores for all specimens.
pc.shapes	A list with the shape coordinates of the extreme ends of all PC axes, e.g. $PC1min
sdev	The standard deviations of the principal components (i.e., the square roots of the eigenvalues of the covariance/correlation matrix, as per prcomp.
rotation	The matrix of variable loadings, as per prcomp.

Author(s)

Dean Adams & Emma Sherratt

Examples

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

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

## To save and use output
PCA <- plotTangentSpace(Y.gpa$coords, groups = gp, legend=TRUE)
summary(PCA)
PCA$pc.shapes
PCA$rotation

##To change colors of groups
col.gp <- rainbow(length(levels(gp)))
   names(col.gp) <- levels(gp)
col.gp <- col.gp[match(gp, names(col.gp))] # col.gp must NOT be a factor
plotTangentSpace(Y.gpa$coords, groups = col.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))
```

`print.advanced.procD.lm` *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'advanced.procD.lm'
print(x, ...)
```

Arguments

`x`	print/summary object (from `advanced.procD.lm`)
`...`	other arguments passed to print/summary

Author(s)

Michael Collyer

print.bilat.symmetry *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from bilat.symmetry)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.compare.pls *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.CR *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from phylo.modularity)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.CR.phylo *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from phylo.modularity)
...	other arguments passed to print/summary

Author(s)

Dean Adams

print.evolrate *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.evolrate1 *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'evolrate1'
print(x, ...)
```

Arguments

x	print/summary object
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.gpagen *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from gpagen)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.morphol.disparity *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from morphol.disparity)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.physignal *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from physignal)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.plotTangentSpace *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.pls *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from phylo.integration or two.b.pls)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.procD.allometry *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from procD.allometry)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.procD.lm *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

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

Arguments

x	print/summary object (from procD.lm)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

print.trajectory.analysis *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'trajectory.analysis'
print(x, angle.type = c("r", "rad", "deg"), ...)
```

Arguments

x	print/summary object
angle.type	Choice between vector correlation or vector angles, in radians or degrees for summarizing results ("r", "rad", "deg", respectively)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

`procD.allometry` *Procrustes ANOVA/regression, specifically for shape-size covariation (allometry)*

Description

Function performs Procrustes ANOVA with permutation procedures to assess statistical hypotheses describing patterns of shape covariation with size for a set of Procrustes-aligned coordinates. Other factors or covariates can also be included in the analysis. This function also provides results for plotting allometric curves.

Usage

```
procD.allometry(f1, f2 = NULL, f3 = NULL, logsz = TRUE, iter = 999,
  seed = NULL, alpha = 0.05, RRPP = TRUE, effect.type = c("F", "SS",
  "cohen"), print.progress = TRUE, data = NULL, ...)
```

Arguments

`f1`	A formula for the relationship of shape and size; e.g., Y ~ X.
`f2`	An optional right-hand formula for the inclusion of groups; e.g., ~ groups.
`f3`	A optional right-hand formula for the inclusion of additional variables; e.g., ~ a + b + c + ...
`logsz`	A logical argument to indicate if the variable for size should be log-transformed.
`iter`	Number of iterations for significance testing
`seed`	An optional argument for setting the seed for random permutations of the re-sampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
`alpha`	The significance level for the homegeneity of slopes test
`RRPP`	A logical value indicating whether residual randomization should be used for significance testing
`effect.type`	One of "F", "SS", or "cohen", to choose from which random distribution to estimate effect size. (The default is "F").
`print.progress`	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.
`data`	A data frame for the function environment, see `geomorph.data.frame`
`...`	Arguments passed on to procD.fit (typically associated with the lm function, such as weights or offset). The function procD.fit can also currently handle either type I, type II, or type III sums of squares and cross-products (SSCP) calculations. Choice of SSCP type can be made with the argument, SS.type; i.e., SS.type = "I" or SS.type = "III". Only advanced users should consider using these additional arguments, as such arguments are experimental in nature.

Details

The function quantifies the relative amount of shape variation attributable to covariation with organism size (allometry) plus other factors in a linear model, plus estimates the probability of this variation ("significance") for a null model, via distributions generated from resampling permutations. Data input is specified by formulae (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 -if the data are based on landmark coordinates - the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with `gpagen`].

There are three formulae that need to be input (see Arguments). The first must contain variables for shape and size, e.g., Y ~ X, where Y (dependent variable) is shape and X (independent variable) is size. The other two formulae are optional to indicate (1) groups for separate allometric curves and (2) additional model variables to consider in the ANOVA. The groups input must be a single factor or multiple factors; e.g., ~ group, or ~ a*b. The resulting ANOVA uses sequential (Type I) sums of squares and cross-products with variables in this order: size, groups (if provided), size*groups (if warranted), other variables (if provided). If a factor for groups is provided, ANOVA for a "homogeneity of slopes" test will also be performed.

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 formulae. Arguments for `lm` can also be passed on via this function. For further information about ANOVA in geomorph, resampling procedures used, and output, see `procD.lm` or `advanced.procD.lm`. If greater flexibility is required for variable order, `advanced.procD.lm` should be used.

It is recommended that `geomorph.data.frame` is used to create and input a data frame. This will reduce problems caused by conflicts between the global and function environments. In the absence of a specified data frame, `procD.allometry` will attempt to coerce input data into a data frame, but success is not guaranteed.

The generic functions, `print`, `summary`, and `plot` all work with `procD.allometry`. The generic function, `plot`, produces plots of allometric curves, using one of three methods input (see below). If diagnostic plots on model residuals are desired, `procD.lm` should be used with the resulting model formula. This, along with the data frame resulting from analysis with `procD.allometry` can be used directly in `procD.lm`, which might be useful for extracting ANOVA components (as `procD.allometry` is far more basic than `procD.lm`, in terms of output).

Notes for geomorph 3.0 and making allometry plots:

Former versions of geomorph had a "plotAllometry" function that performed ANOVA and produced plots of allometry curves. In geomorph 3.0, the `plot` function is used with `procD.allometry` objects to produce such plots. The following arguments can be used in `plot` to achieve desired results.

- method = ("CAC, "RegScore, "PredLine"). Choose the desired plot method.
- warpgrids: default = TRUE. Logical value to indicate whether warpgrids should be plotted. (Only workds with 3D array data)
- label: can be logical to label points (1:n) - e.g., label = TRUE - or a vector indicating text to use as labels.
- mesh: A mesh3d object to be warped to represent shape deformation of the minimum and maximum size if warpgrids=TRUE (see `warpRefMesh`).

Use ?plot.procD.allometry to understand the arguments used. The following are brief descriptions of the different plotting methods using plot, with references.

- 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.
- 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).
- 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).

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Value

An object of class "procD.allometry" is a list containing the following:

HOS.test	ANOVA for a homogeneity of slopes test (if groups are provided).
aov.table	An analysis of variance table, based on inputs and the homogenetiy of slopes test.
alpha	The significance level criterion for the homogeneity of slopes test.
perm.method	A value indicating whether "RRPP" or randomization of "raw" vales was used.
permutations	The number of random permutations used in the resampling procedure.
data	The data frame for the model.
random.SS	A matrix or vector of random SS found via the resampling procedure used.
random.F	A matrix or vector of random F values found via the resampling procedure used.
random.cohenf	A matrix or vector of random Cohen's f-squared values found via the resampling procedure used.
call	The matched call.
formula	The resulting formula, which can be used in follow-up analyses. Irrespective of input, shape = Y in the formula, and the variable used for size is called "size".
CAC	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.

RSC	The residual shape component (associated with CAC approach)
Reg.proj	The projected regression scores on the regression of shape on size. For a single group, these shape scores are mathematically identical to the CAC (Adams et al. 2013).
pred.val	Principal component scores (first PC) of predicted values.
ref	the reference configuration (if input coordinates are in a 3D array).
gps	A vector of group names.
size	A vector of size scores.
logsz	A logical value to indicate if size values were log=transformed for analysis.
A	Procrustes (aligned) residuals.
Ahat	Predicted Procrustes residuals(matching array or matrix, as input).
Ahat.at.min	Predicted Procrustes residuals, specifically at minimum size.
Ahat.at.max	Predicted Procrustes residuals, specifically at maximum size.
p	landmark number
k	landmark dimensions

Author(s)

Michael Collyer

References

Adams, D.C., F.J. Rohlf, and D.E. Slice. 2013. A field comes of age: geometric morphometrics in the 21st century. Hystrix. 24:7-14.

Adams, D. C., and A. Nistri. 2010. Ontogenetic convergence and evolution of foot morphology in European cave salamanders (Family: Plethodontidae). BMC Evol. Biol. 10:1-10.

Drake, A. G., and C. P. Klingenberg. 2008. The pace of morphological change: Historical transformation of skull shape in St Bernard dogs. Proc. R. Soc. B. 275:71-76.

Mitteroecker, P., P. Gunz, M. Bernhard, K. Schaefer, and F. L. Bookstein. 2004. Comparison of cranial ontogenetic trajectories among great apes and humans. J. Hum. Evol. 46:679-698.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

See Also

procD.lm and advanced.procD.lm within geomorph; lm for more on linear model fits

Examples

```
# Simple allometry
data(plethodon)
Y.gpa <- gpagen(plethodon$land)    #GPA-alignment
gdf <- geomorph.data.frame(Y.gpa, site = plethodon$site,
species = plethodon$species) # geomorph data frame
plethAllometry <- procD.allometry(coords~Csize, f2 = NULL, f3=NULL,
logsz = TRUE, data=gdf, iter=499)
summary(plethAllometry)
plot(plethAllometry, method = "PredLine")
plot(plethAllometry, method = "RegScore")

## Obtaining size-adjusted residuals (and allometry-free shapes)
plethAnova <- procD.lm(plethAllometry$formula,
     data = plethAllometry$data, iter = 499, RRPP=TRUE)
summary(plethAnova) # same ANOVA Table
shape.resid <- arrayspecs(plethAnova$residuals,
   p=dim(Y.gpa$coords)[1], k=dim(Y.gpa$coords)[2]) # size-adjusted residuals
adj.shape <- shape.resid + array(Y.gpa$consensus, dim(shape.resid)) # allometry-free shapes
plotTangentSpace(adj.shape) # PCA of allometry-free shape

# Group Allometries
plethAllometry <- procD.allometry(coords~Csize, ~species*site,
logsz = TRUE, data=gdf, iter=499, RRPP=TRUE)
summary(plethAllometry)
plot(plethAllometry, method = "PredLine")

# Using procD.lm to perform diagnostic residual plots
plethANOVA <- procD.lm(plethAllometry$formula,
data = plethAllometry$data, iter = 499, RRPP=TRUE)
summary(plethANOVA) # Same ANOVA
plot(plethANOVA) # diagnostic plot instead of allometry plot
```

`procD.lm` *Procrustes ANOVA/regression for shape data*

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, seed = NULL, RRPP = TRUE, effect.type = c("F",
  "SS", "cohen"), int.first = FALSE, data = NULL, print.progress = TRUE,
  ...)
```

Arguments

f1	A formula for the linear model (e.g., y~x1+x2)
iter	Number of iterations for significance testing
seed	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
RRPP	A logical value indicating whether residual randomization should be used for significance testing
effect.type	One of "F", "SS", or "cohen", to choose from which random distribution to estimate effect size. (The default, "cohen", is for Cohen's f-squared values. Values are log-transformed before z-score calculation to assure normally distributed data.)
int.first	A logical value to indicate if interactions of first main effects should precede subsequent main effects
data	A data frame for the function environment, see geomorph.data.frame
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.
...	Arguments passed on to procD.fit (typically associated with the lm function, such as weights or offset). The function procD.fit can also currently handle either type I, type II, or type III sums of squares and cross-products (SSCP) calculations. Choice of SSCP type can be made with the argument, SS.type; i.e., SS.type = "I" or SS.type = "III". Only advanced users should consider using these additional arguments, as such arguments are experimental in nature.

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 -if the data based on landmark coordinates - 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. It is also recommended that geomorph.data.frame is used to create and input a data frame. This will reduce problems caused by conflicts between the global and function

environments. In the absence of a specified data frame, procD.lm will attempt to coerce input data into a data frame, but success is not guaranteed.

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. 2015). 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 either the SS, F, or Cohen's f-squared sampling distributions generated, which might be more intuitive for P-values than F-values (see Collyer et al. 2015). Values from these distributions are log-transformed prior to effect size estimation, to assure normally distributed data. The SS type will influence how Cohen's f-squared values are calculated. Cohen's f-squared values are based on partial eta-squared values that can be calculated sequentially or marginally, as with SS.

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.)

The generic functions, `print`, `summary`, and `plot` all work with `procD.lm`. The generic function, `plot` has several options for plotting, using `plot.procD.lm`. Diagnostics plots, principal component plots (rotated to first PC of covariance matrix of fitted values), and regression plots can be performed. The latter is fundamentally similar to the plotting options for `procD.allometry`. One must provide a linear predictor, and can choose among common regression component (CRC), predicted values (PredLine), or regression scores (RegScore). See `procD.allometry` for details. In these plotting optons, the predictor does not need to be size, and fitted values and residuals from the procD.lm fit are used rather than mean-centered values.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Value

An object of class "procD.lm" is a list containing the following

aov.table	An analysis of variance table; the same as the summary.
call	The matched call.
coefficients	A vector or matrix of linear model coefficients.
Y	The response data, in matrix form.
X	The model matrix.
QR	The QR decompositions of the model matrix.
fitted	The fitted values.
residuals	The residuals (observed responses - fitted responses).
weights	The weights used in weighted least-squares fitting. If no weights are used, NULL is returned.
Terms	The results of the terms function applied to the model matrix
term.labels	The terms used in constructing the aov.table.
data	The data frame for the model.
SS	The sums of squares for each term, model residuals, and the total.
SS.type	The type of sums of squares. One of type I or type III.
df	The degrees of freedom for each SS.
R2	The coefficient of determination for each model term.
F	The F values for each model term.
permutations	The number of random permutations (including observed) used.
random.SS	A matrix or vector of random SS found via the resampling procedure used.
random.F	A matrix or vector of random F values found via the resampling procedure used.
random.cohenf	A matrix or vector of random Cohen's f-squared values found via the resampling procedure used.
permutations	The number of random permutations (including observed) used.
effect.type	The distribution used to estimate effect-size.
perm.method	A value indicating whether "Raw" values were shuffled or "RRPP" performed.

Author(s)

Dean Adams and Michael Collyer

References

Anderson MJ. 2001. A new method for non-parametric multivariate analysis of variance. Austral Ecology 26: 32-46.

Anderson MJ. and C.J.F. terBraak. 2003. Permutation tests for multi-factorial analysis of variance. Journal of Statistical Computation and Simulation 73: 85-113.

Goodall, C.R. 1991. Procrustes methods in the statistical analysis of shape. Journal of the Royal Statistical Society B 53:285-339.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

See Also

advanced.procD.lm, procD.pgls, and nested.update within geomorph; lm for more on linear model fits.

Examples

```
### MANOVA example for Goodall's F test (multivariate shape vs. factors)
data(plethodon)
Y.gpa <- gpagen(plethodon$land)    #GPA-alignment
gdf <- geomorph.data.frame(shape = Y.gpa$coords,
site = plethodon$site, species = plethodon$species) # geomorph data frame

procD.lm(shape ~ species * site, data = gdf, iter = 999, RRPP = FALSE) # randomize raw values
procD.lm(shape ~ species * site, data = gdf, iter = 999, RRPP = TRUE) # randomize residuals

### Regression example
data(ratland)
rat.gpa<-gpagen(ratland)         #GPA-alignment
gdf <- geomorph.data.frame(rat.gpa) # geomorph data frame is easy without additional input

procD.lm(coords ~ Csize, data = gdf, iter = 999, RRPP = FALSE) # randomize raw values
procD.lm(coords ~ Csize, data = gdf, iter = 999, RRPP = TRUE) # randomize raw values
# Outcomes should be exactly the same

### Extracting objects and plotting options
rat.anova <- procD.lm(coords ~ Csize, data = gdf, iter = 999, RRPP = TRUE)
summary(rat.anova)
# diagnostic plots
plot(rat.anova, type = "diagnostics")
# diagnostic plots, including plotOutliers
plot(rat.anova, type = "diagnostics", outliers = TRUE)
# PC plot rotated to major axis of fitted values
plot(rat.anova, type = "PC", pch = 19, col = "blue")
# Uses residuals from model to find the commonom regression component
# for a predictor from the model
plot(rat.anova, type = "regression", predictor = gdf$Csize, reg.type = "CRC",
pch = 19, col = "green")
# Uses residuals from model to find the projected regression scores
rat.plot <- plot(rat.anova, type = "regression", predictor = gdf$Csize, reg.type = "RegScore",
pch = 21, bg = "yellow")

# TPS grids for min and max scores in previous plot
preds <- shape.predictor(gdf$coords, x = rat.plot$RegScore,
```

```
                        predmin = min(rat.plot$RegScore),
                        predmax = max(rat.plot$RegScore))
M <- rat.gpa$consensus
plotRefToTarget(M, preds$predmin, mag=3)
plotRefToTarget(M, preds$predmax, mag=3)

attributes(rat.anova)
rat.anova$fitted # just the fitted values
```

`procD.pgls` *Phylogenetic ANOVA/regression for shape data*

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

```
procD.pgls(f1, phy, iter = 999, seed = NULL, int.first = FALSE,
  effect.type = c("F", "cohen"), RRPP = TRUE, data = NULL,
  print.progress = TRUE, ...)
```

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
`seed`	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
`int.first`	A logical value to indicate if interactions of first main effects should precede subsequent main effects
`effect.type`	One of "F" or "ochen", to choose from which random distribution to estimate effect size. (The default is "F". Values are log-transformed before z-score calculation to assure normally distributed effect sizes.)
`RRPP`	a logical value indicating whether residual randomization should be used for significance testing
`data`	A data frame for the function environment, see `geomorph.data.frame`
`print.progress`	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

... Arguments passed on to procD.fit (typically associated with the lm function, such as weights or offset). The function procD.fit can also currently handle either type I, type II, or type III sums of squares and cross-products (SSCP) calculations. Choice of SSCP type can be made with the argument, SS.type; i.e., SS.type = "I" or SS.type = "III". Only advanced users should consider using these additional arguments, as such arguments are experimental in nature.

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 `gpagen`]. 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 user must also specify a phylogeny describing the evolutionary relationships among species (of class phylo). Note that the specimen labels for both X and y must match the labels on the tips of the phylogeny.

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. It is also recommended that `geomorph.data.frame` is used to create and input a data frame. This will reduce problems caused by conflicts between the global and function environments. In the absence of a specified data frame, procD.pgls will attempt to coerce input data into a data frame, but success is not guaranteed.

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 all estimates of statistical values are obtained for the permuted data, which are compared to the observed value to assess significance. This approach has been shown to have appropriate type I error rates, whereas an alternative procedure for phylogenetic regression of morphometric shape data displays elevated type I error rates (see Adams and Collyer 2015).

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. 2015). 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 either the F or Cohen's f-squared sampling distributions generated, which might be more intuitive for P-values than F-values (see Collyer et al. 2015). Values from these distributions are log-transformed prior to effect size estimation,

to assure normally distributed data. The SS type will influence how Cohen's f-squared values are calculated. Cohen's f-squared values are based on partial eta-squared values that can be calculated sequentially or marginally, as with SS.

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.)

The generic functions, `print`, `summary`, and `plot` all work with `procD.pgls`. The generic function, `plot`, produces diagnostic plots for Procrustes residuals of the linear fit.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Value

procD.lm.pgls returns an object of class "procD.lm". See `procD.lm` for a description of the list of results generated. Additionally, procD.pgls provides the phylogenetic correction matrix, Pcor, plus "pgls" adjusted coefficients, fitted values, residuals, and mean.

Author(s)

Dean Adams and Michael Collyer

References

Adams, D.C. 2014. A method for assessing phylogenetic least squares models for shape and other high-dimensional multivariate data. Evolution. 68:2675-2688.

Adams, D.C., and M.L. Collyer. 2015. Permutation tests for phylogenetic comparative analyses of high-dimensional shape data: what you shuffle matters. Evolution. 69:823-829.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

Examples

```
### Example of D-PGLS for high-dimensional data
data(plethspecies)
Y.gpa<-gpagen(plethspecies$land)    #GPA-alignment
gdf <- geomorph.data.frame(Y.gpa, phy = plethspecies$phy)
procD.pgls(coords ~ Csize, phy = phy, data = gdf, iter = 999, RRPP = FALSE) # randomize raw values
procD.pgls(coords ~ Csize, phy = phy, data = gdf, iter = 999, RRPP = TRUE) # randomize residuals

### Extracting objects
pleth.pgls <- procD.pgls(coords ~ Csize, phy = phy, data = gdf, iter = 999, RRPP = TRUE)
summary(pleth.pgls)
plot(pleth.pgls)
pleth.pgls$Pcor # the phylogenetic transformation (correction) matrix
```

---

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).

Details

These data were previously aligned with GPA. Centroid size (CS) is also provided.

Author(s)

Michael Collyer

References

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 113: doi:10.1038/hdy.2014.75.

ratland *Landmark data from dataset rat*

Description

Landmark data from dataset rat

Author(s)

Dean Adams

References

Bookstein, F. L. 1991. Morphometric tools for landmark data: Geometry and Biology. Cambridge Univ. Press, New York.

read.morphologika *Read landmark data from Morphologika file(s)*

Description

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

Usage

```
read.morphologika(filelist)
```

Arguments

filelist	The name of a Morphologika *.txt file containing two- or three-dimensional landmark data. Alternatively, a character vector of names of Morphologika *.txt file (such as made using list.files)

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 multiple morphologika files are specified (containing the same number of landmarks for all specimens), function returns a single object for files.

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'). If the header "[polygon]" is present, then a matrix of the landmark addresses for the polygon wireframe is returned (see polygon3d or polygon).

NOTE: For multiple morphologika files that each contain only a single specimen (such as those exported from Stratovan Checkpoint software), one can add specimen names to the returned 3D array by: dimnames(mydata)[[3]] <- gsub (".txt", "", filelist)).

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

O'Higgins P and Jones N (1998) Facial growth in Cercocebus torquatus: An application of three dimensional geometric morphometric techniques to the study of morphological variation. Journal of Anatomy. 193: 251-272

O'Higgins P and Jones N (2006) Tools for statistical shape analysis. Hull York Medical School.

---

read.ply *Read mesh data (vertices and faces) from ply files*

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$ply
myply$material <- "gray" # using color word
myply$material <- "#FCE6C9" # using RGB code
```

---

readland.nts *Read landmark data matrix from nts file*

Description

Read single *.nts file containing landmark coordinates for a set of specimens

Usage

```
readland.nts(file)
```

Arguments

file	the name of a *.nts file containing two- or three-dimensional landmark data to be read in

Details

Function reads a single *.nts file containing two- or three-dimensional landmark coordinates for multiple specimens.

This is for NTS files of the "multiple specimen format" (details below), which is not the same as readmulti.nts.

NTS files are text files in one of the standard formats for geometric morphometrics (see Rohlf 2012). Multiple specimen format: 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.

Special NTS files: *.dta files in the written by IDAV Landmark Editor, and *.nts files written by Stratovan Checkpoint 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. Users must change manually the header (in a text editor) before using this function

Value

Function returns a 3D array (p x k x n), 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 names in the *.nts file (if included).

Author(s)

Dean Adams & Emma Sherratt

References

Rohlf, F. J. 2012 NTSYSpc: Numerical taxonomy and multivariate analysis system. Version 2.2. Exeter Software, New York.

See Also

readmulti.nts

---

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 = TRUE)
```

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.

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 *.tps file.

Author(s)

Dean Adams & Emma Sherratt

References

Rohlf, F. J. 2010. tpsRelw: Relative warps analysis. Version 1.49. Department of Ecology and Evolution, State University of New York at Stony Brook, Stony Brook, NY.

`readmulti.nts` *Read landmark data from multiple nts files*

Description

Read multiple *.nts files, each containing landmark coordinates for a single specimen

Usage

```
readmulti.nts(filelist)
```

Arguments

`filelist`	a character vector of file names for the *.nts files to be read in

Details

Function reads a character vector of filenames for a set of *.nts files that each contain two- or three-dimensional landmark coordinates for a single specimen (e.g., exported from `digit.fixed` and `digitsurface`).

This is for NTS files of the "single specimen format" (details below), which is not the same as `readland.nts`.

NTS files are text files in one of the standard formats for geometric morphometrics (see Rohlf 2012). Single specimen format: 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 landmarks (p) and the dimensions (k) of 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 3D array (p x k x n), 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 file names.

Author(s)

Dean Adams & Emma Sherratt

References

Rohlf, F. J. 2012 NTSYSpc: Numerical taxonomy and multivariate analysis system. Version 2.2. Exeter Software, New York.

See Also

readland.nts

---

scallopPLY *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

Serb et al. (2011). "Morphological convergence of shell shape in distantly related scallop species (Mollusca: Pectinidae)." Zoological Journal of the Linnean Society 163: 571-584.

---

scallops *Landmark data from scallop shells*

Description

Landmark data from scallop shells

Author(s)

Dean Adams and Erik Otarola-Castillo

References

Serb et al. (2011). "Morphological convergence of shell shape in distantly related scallop species (Mollusca: Pectinidae)." Zoological Journal of the Linnean Society 163: 571-584.

shape.predictor *Shape prediction from numeric predictors*

Description

Function estimates one or more configurations based on one or more linear predictors, such as PC scores allometric relationships, or any other least squares or partial least squares regression. These configurations can be used with plotRefToTarget to generate graphical representations of shape change, based on prediction criteria.

Usage

```
shape.predictor(A, x = NULL, Intercept = FALSE, method = c("LS", "PLS"),
  ...)
```

Arguments

A	A 3D array (p x k x n) containing Procrustes residuals either from GPA or fitted values from a previous analytical procedure.
x	Linear (numeric) predictors. Can be a vector or a matrix, or a list containing vectors or matrices. Values must be numeric. If a factor is desired, one should use model.matrix to obtain a design matrix. This will impact how prediction criteria need to be provided (see below).
Intercept	Logical value to indicate whether an intercept should be used in the linear equation for predictions. Generally, this value will be FALSE for shape predictions made in ordination plots. It should be TRUE in cases where the expected shape at the point the predictor has a value of 0 is not the mean shape.
method	A choice between least squares (LS) or partial least squares (PLS) regression for prediction. The function defaults to LS prediction. PLS might be chosen in cases where correlation is preferred over linear regression. If PLS is chosen, a two-block PLS analysis using two.b.pls should be performed first, as only the first singular vector for predictors will be used for defining prediction criteria (see below).
...	Any number of prediction criteria. Criteria should be presented as either a scalar (if one predictor is provided) or a vector (if more than one predictor or a prediction matrix is provided); e.g., pred1 = c(0.1, -0.5), pred2 = c(-0.2, -0.1) (which would be the case if two predictors were provided). It is essential that the number of elements in any prediction criterion matches the number of predictors. Caution should be used when providing a design matrix to ensure that correct dummy variables are used in prediction criteria, and that either 1) an intercept is not included in the design and 2) is TRUE in the Intercept argument; or or 1) an intercept is included in the design and 2) is FALSE in the Intercept argument; or 1) an intercept is not included in the design and 2) is FALSE in the Intercept argument, if no intercept is desired.

Value

A list of predicted shapes matching the number of vectors of prediction criteria provides. The predications also have names matching those of the prediction criteria.

Author(s)

Michael Collyer

Examples

```
# Examples using Plethodon data

data("plethodon")

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

preds <- shape.predictor(Y.gpa$coords, x= NULL, Intercept = FALSE,
pred1 = -0.1, pred2 = 0.1) # PC 1 extremes, sort of
M <- mshape(Y.gpa$coords)
plotRefToTarget(M, preds$pred1)
plotRefToTarget(M, preds[[1]]) # same result
plotRefToTarget(M, preds$pred2)

PCA <- plotTangentSpace(Y.gpa$coords)
PC <- PCA$pc.scores[,1]
preds <- shape.predictor(Y.gpa$coords, x= PC, Intercept = FALSE,
pred1 = min(PC), pred2 = max(PC)) # PC 1 extremes, more technically
plotRefToTarget(M, preds$pred1)
plotRefToTarget(M, preds$pred2)

PC <- PCA$pc.scores[,1:2]
# user-picked spots - can be anything, but it in this case, apparent groups
preds <- shape.predictor(Y.gpa$coords, x= PC, Intercept = FALSE,
                   pred1 = c(0.045,-0.02), pred2 = c(-0.025,0.06), pred3 = c(-0.06,-0.04))
plotRefToTarget(M, preds$pred1)
plotRefToTarget(M, preds$pred2)
plotRefToTarget(M, preds$pred3)

# allometry example - straight-up allometry

preds <- shape.predictor(Y.gpa$coords, x= log(Y.gpa$Csize), Intercept = TRUE,
                       predmin = min(log(Y.gpa$Csize)), predmax = max(log(Y.gpa$Csize)))

plotRefToTarget(M, preds$predmin, mag=3)
plotRefToTarget(M, preds$predmax, mag=3)

# allometry example - using RegScore or PredLine via procD.allometry

gdf <- geomorph.data.frame(Y.gpa)
plethAllometry <- procD.allometry(coords~Csize, data=gdf, logsz = TRUE)
plot(plethAllometry, method="RegScore")
```

```
preds <- shape.predictor(plethAllometry$Ahat, x= plethAllometry$Reg.proj, Intercept = FALSE,
                         predmin = min(plethAllometry$Reg.proj),
                         predmax = max(plethAllometry$Reg.proj))
plotRefToTarget(M, preds$predmin, mag=3)
plotRefToTarget(M, preds$predmax, mag=3)

plot(plethAllometry, method="PredLine")
preds <- shape.predictor(plethAllometry$Ahat, x= plethAllometry$pred.val, Intercept = FALSE,
                         predmin = min(plethAllometry$pred.val),
                         predmax = max(plethAllometry$pred.val))
plotRefToTarget(M, preds$predmin, mag=3)
plotRefToTarget(M, preds$predmax, mag=3)

# using factors via PCA

gdf <- geomorph.data.frame(Y.gpa, species = plethodon$species, site = plethodon$site)
pleth <- procD.lm(coords ~ species*site, data=gdf)
PCA <- prcomp(pleth$fitted)
plot(PCA$x, asp=1, pch=19)

means <- unique(round(PCA$x,3))
means # note: suggests 3 PCs useful enough

preds <- shape.predictor(arrayspecs(pleth$fitted, 12,2), x= PCA$x[,1:3],
                         Intercept = FALSE,
                         pred1 = means[1,1:3],
                         pred2 = means[2,1:3],
                         pred3 = means[3,1:3],
                         pred4 = means[4,1:3])
plotRefToTarget(M, preds$pred1, mag=2)
plotRefToTarget(M, preds$pred2, mag=2)
plotRefToTarget(M, preds$pred3, mag=2)
plotRefToTarget(M, preds$pred4, mag=2)

# Using a design matrix for factors

X <- pleth$X
X # includes intercept; remove for better functioning
X <- X[,-1]
symJord <- c(0,1,0) # design for P. Jordani in sympatry
alloJord <- c(0,0,0) # design for P. Jordani in allopatry
preds <- shape.predictor(arrayspecs(pleth$fitted, 12,2), x = X, Intercept = TRUE,
                         symJord=symJord, alloJord=alloJord)
plotRefToTarget(M, preds$symJord, mag=2)
plotRefToTarget(M, preds$alloJord, mag=2)

# PLS Example

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

# 2B-PLS between head shape and food use data
PLS <-two.b.pls(plethShapeFood$food,Y.gpa$coords, iter=999)
```

```
summary(PLS)
plot(PLS)

preds <- shape.predictor(Y.gpa$coords, plethShapeFood$food, Intercept = FALSE,
                         method = "PLS",
                         pred1 = 2, pred2 = -4, pred3 = 2.5) # using PLS plot as a guide
M <- mshape(Y.gpa$coords)
plotRefToTarget(M, preds$pred1, mag=2)
plotRefToTarget(M, preds$pred2, mag=2)
plotRefToTarget(M, preds$pred3, mag=2)
```

---

summary.advanced.procD.lm

Print/Summary Function for geomorph

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'advanced.procD.lm'
summary(object, ...)
```

Arguments

object	print/summary object (from advanced.procD.lm)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

---

summary.bilat.symmetry

Print/Summary Function for geomorph

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'bilat.symmetry'
summary(object, ...)
```

Arguments

object	print/summary object (from bilat.symmetry)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

---

summary.compare.pls *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'compare.pls'
summary(object, ...)
```

Arguments

object	print/summary object
...	other arguments passed to print/summary

Author(s)

Michael Collyer

---

summary.CR *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'CR'
summary(object, ...)
```

Arguments

object	print/summary object (from phylo.modularity)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

`summary.CR.phylo` *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'CR.phylo'
summary(object, ...)
```

Arguments

`object`	print/summary object (from `phylo.modularity`)
`...`	other arguments passed to print/summary

Author(s)

Dean Adams

`summary.evolrate` *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'evolrate'
summary(object, ...)
```

Arguments

`object`	print/summary object
`...`	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.evolrate1 *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'evolrate1'
summary(object, ...)
```

Arguments

object	print/summary object
...	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.gpagen *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'gpagen'
summary(object, ...)
```

Arguments

object	print/summary object (from gpagen)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.morphol.disparity *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'morphol.disparity'
summary(object, ...)
```

Arguments

object	print/summary object (from morphol.disparity)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.physignal *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'physignal'
summary(object, ...)
```

Arguments

object	print/summary object (from physignal)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.plotTangentSpace *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'plotTangentSpace'
summary(object, ...)
```

Arguments

object	print/summary object
...	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.pls *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'pls'
summary(object, ...)
```

Arguments

object	print/summary object (from phylo.integration or two.b.pls)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.procD.allometry *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'procD.allometry'
summary(object, ...)
```

Arguments

object	print/summary object (from procD.allometry)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.procD.lm *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'procD.lm'
summary(object, ...)
```

Arguments

object	print/summary object (from procD.lm)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

summary.trajectory.analysis *Print/Summary Function for geomorph*

Description

Print/Summary Function for geomorph

Usage

```
## S3 method for class 'trajectory.analysis'
summary(object, angle.type = c("r", "rad",
  "deg"), ...)
```

Arguments

object	print/summary object
angle.type	Choice between vector correlation or vector angles, in radians or degrees for summarizing results ("r", "rad", "deg", respectively)
...	other arguments passed to print/summary

Author(s)

Michael Collyer

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, f2 = NULL, iter = 999, seed = NULL,
  traj.pts = NULL, data = NULL, print.progress = TRUE, ...)
```

Arguments

f1	A formula for the linear model, for trajectories (e.g., Y ~ A or Y ~ A * B). The right hand side of this formula can contain only one or two factors.
f2	A formula for additional covariates (e.g., ~ x1 + x2)
iter	Number of iterations for significance testing
seed	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
traj.pts	An optional value specifying the number of points in each trajectory (if f1 contains a single factor)
data	A data frame for the function environment, see geomorph.data.frame
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.
...	Arguments passed on to procD.fit (typically associated with the lm function, such as weights or offset). The function procD.fit can also currently handle either type I, type II, or type III sums of squares and cross-products (SSCP) calculations. Choice of SSCP type can be made with the argument, SS.type; i.e., SS.type = "I" or SS.type = "III". Only advanced users should consider using these additional arguments, as such arguments are experimental in nature.

Details

The function quantifies phenotypic shape change trajectories from a set of specimens, and assesses variation in attributes of the trajectories 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 two formulae (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'. It is also assumed that the landmarks have previously been aligned using Generalized Procrustes Analysis (GPA) [e.g., with gpagen]. 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 first formula, f1, must contain the independent variable on the left-hand side of the formula (e.g., Y ~) and either a single factor or a two factor interaction on the right-hand side. If a single factor is provided, e.g., Y ~ A, it is assumed that groups to be described are the levels of factor A and that the data in Y comprise trajectories. In this case, the traj.pts = NULL argument must be changed to a numeric value to define the number of points in the trajectory. It is also assumed that the data are structured as variables within points. For example, y11 y21 y31 y12 y22 y32 y13 y23 y33 y14 y24 y34 would be columns of a matrix, Y, describing

a 4-point trajectory in a data space defined by three variables. This is the proper arrangement; the following is an improper arrangement: y11 y12 y13 y14 y21 y22 y23 y24 y31 y32 y33 y34, as it groups points within variables. This approach is typical when comparing motion paths (see Adams and Cerney 2007).

If f1 is a two-factor factorial model, e.g., Y ~ A*B, it is assumed that the first factor defines groups, the second factor defines trajectory points, and that trajectories are to be estimated from the linear model. In this case, the preceding example would have a Y matrix comprised only of y1, y2, and y3, but the factor B would contain levels to define the four points (see Examples).

If one wishes to include other variables in the linear model, they should be indicated in the second formula, f2. This formula can be simply a right-hand formula, e.g., ~ x1 + x2 + x3 +... Variables in this formula will typically be covariates that one wishes to include to account for extraneous sources of shape variation. An analysis of variance (ANOVA) will be performed with type I sums of squares (SS) and a randomized residual permutation procedure (RRPP). The variables in f2 will be added prior to the trajectory defining variables in f1.

Once the function has performed the analysis, a plot can be generated 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, green, blue, cyan, magenta, yellow, and gray. However, one can override these colors with the argument, "group.cols" in plots using the function, `plot`. This will change the trajectory line colors. One can also override default initial-middle-end point colors with the argument, "pt.seq.pattern". The default is c("white", "gray", "black") for gray points, but with white initial points and black end points. If changed, the pt.seq.pattern argument must be a vector with three color values. One can also uniformly vary the size of points with the argument, "pt.scale". Examples are provided below.

The function, `summary` can be used to provide an ANOVA summary plus pairwise statistics of a an object of class "trajectory.analysis". The argument, angle.type = c("r", "rad", "deg") can be used to toggle between vector correlations, vector angles in radians, or vector angles in degrees, respectively.

Notes for geomorph 3.0:

Previous versions of geomorph had two separate analytical approaches based on whether trajectories were estimated or provided (as might be the case with motion trajectories; see Adams and Cerney 2007). Starting with geomorph 3.0, commensurate analytical approaches are used. This involves converting 1 x vp vectors for trajectpries, were p is the number of trajectory points and v is the number of variables in the data space, into p x v matrices, analogous to the procedure for estimating trajectories. Thus, rather than providing separate ANOVAs for size, orientation, and shape of trajectories, a general ANOVA is provided with pairwise statistics for the same attribute differences. This change does not compromise any interpretations made with previous versions of geomorph, but enhances inferential capacity by providing pairwise statistics and P-values.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and

subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Value

An object of class "trajectory.analysis" returns a list of the following:

`aov.table` Procrustes ANOVA table.

`means` The observed least squares means based on the linear model.

`pc.means` The observed least squares means rotated to their principal components.

`pc.data` The observed data rotated to the principal components calculated from the covariance matrix among means. In the case that trajectories are input as data, pc.data is a matrix of the trajectories rotated to align with principal axes in the data space.

`pc.summary` A table summarizing the percent variation explained by each pc axis, equivalent to summary of `prcomp`.

`pc.trajectories`
The observed trajectories rotated to the principal components calculated from the covariance matrix among means. In the case that trajectories are input as data, pc.trajectories is a list of the the rows of pc.data as separate matrices.

`random.means` A list of matrices of means calculated in the random permutations.

`random.trajectories`
A list of all random means reconfigured as trajectories. The observed case is the first random set.

`path.distances` The path distances of each observed trajectory.

`magnitude.diff` A matrix of the absolute differences in pairwise path distances.

`trajectory.cor` A matrix of pairwise vector correlations among trajectory PCs

`trajectory.angle.rad`
The vector correlations transformed into angles, in radians.

`trajectory.angle.deg`
The vector correlations transformed into angles, in degrees.

`trajectory.shape.dist`
A matrix of pairwise Procrustes distances among trajectory shapes.

`P.magnitude.diff`
P-values corresponding to trajectory magnitude differences.

`P.angle.diff` P-values corresponding to angular differences in trajectories.

`P.shape.diff` P-values corresponding to trajectory shape differences.

`Z.magnitude.diff`
Effect sizes of observed trajectory magnitude differences.

`Z.angle.diff` Effect sizes of observed angular differences in trajectories.

`Z.shape.diff` Effect sizes of observed trajectory shape differences.

call	The matched call.
groups	Factor representing group names for subsequent plotting.
permutations	The number of random permutations used in the RRPP applied to the ANOVA and trajectory statistics.
trajectory.type	A value of 1 if trajectories were provided or 2 if they were estimated.

Author(s)

Dean Adams and Michael Collyer

References

Collyer, M.L., and D.C. Adams. 2013. Phenotypic trajectory analysis: Comparison of shape change patterns in evolution and ecology. Hystrix. 24:75-83.

Adams, D. C. 2010. Parallel evolution of character displacement driven by competitive selection in terrestrial salamanders. BMC Evol. Biol. 10:1-10.

Adams, D. C., and M. M. Cerney. 2007. Quantifying biomechanical motion using Procrustes motion analysis. J. Biomech. 40:437-444.

Adams, D. C., and M. L. Collyer. 2007. The analysis of character divergence along environmental gradients and other covariates. Evolution 61:510-515.

Adams, D. C., and M. L. Collyer. 2009. A general framework for the analysis of phenotypic trajectories in evolutionary studies. Evolution 63:1143-1154.

Collyer, M. L., and D. C. Adams. 2007. Analysis of two-state multivariate phenotypic change in ecological studies. Ecology 88:683-692.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

Examples

```
# Estimate trajectories from LS means in 2-factor model

data(plethodon)
Y.gpa <- gpagen(plethodon$land)
gdf <- geomorph.data.frame(Y.gpa, species = plethodon$species, site = plethodon$site)

TA <- trajectory.analysis(coords ~ species*site, data=gdf, iter=199)
summary(TA, angle.type = "deg")
plot(TA)

# Change order of groups

site <- as.factor(plethodon$site)
levels(site) <- c("Symp", "Allo")
gdf <- geomorph.data.frame(Y.gpa, species = plethodon$species, site = site)
```

```
TA <- trajectory.analysis(coords ~ species*site, data=gdf, iter=199)
summary(TA, angle.type = "deg")
plot(TA)

attributes(TA) # list of extractable parts

# Add Centroid size as a covariate

TA <- trajectory.analysis(f1 = coords ~ species*site, f2 = ~ Csize, data=gdf, iter=199)
summary(TA, angle.type = "deg")
plot(TA)

# Change trajectory colors in plot
plot(TA, group.cols = c("dark red", "dark blue"))

# Change size of points and lines
plot(TA, group.cols = c("dark red", "dark blue"), pt.scale=1.5)

# Motion paths represented by 5 time points per motion

data(motionpaths)

gdf <- geomorph.data.frame(trajectories = motionpaths$trajectories,
groups = motionpaths$groups)
TA <- trajectory.analysis(f1 = trajectories ~ groups,
traj.pts = 5, data=gdf, iter=199)
summary(TA)
plot(TA)
plot(TA, group.cols = c("dark red", "dark blue", "dark green", "yellow"), pt.scale = 1.3)
plot(TA, group.cols = c("dark red", "dark blue", "dark green", "yellow"),
pt.seq.pattern = c("green", "gray30", "red"), pt.scale = 1.3)
```

`two.b.pls` *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 to blocks of Procrustes-aligned coordinates (or other variables)

Usage

```
two.b.pls(A1, A2, iter = 999, seed = NULL, print.progress = TRUE)
```

Arguments

`A1`	A 3D array (p x k x n) containing GPA-aligned coordinates for the first block, or a matrix (n x variables)

A2	A 3D array (p x k x n) containing GPA-aligned coordinates for the second block, or a matrix (n x variables)
iter	Number of iterations for significance testing
seed	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If seed = "random", a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
print.progress	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

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). It is also assumed that the separate inputs have specimens (observations) in the same order.

The generic functions, print, summary, and plot all work with two.b.pls. The generic function, plot, produces a two-block.pls plot. This function calls plot.pls, which has two additional arguments (with defaults): label = NULL, warpgrids = TRUE. These arguments allow one to include a vector to label points and a logical statement to include warpgrids, respectively. Warpgrids can only be included for 3D arrays of Procrustes residuals. The plot is a plot of PLS scores from Block1 versus Block2 performed for the first set of PLS axes.

For more than two blocks:

If one wishes to consider 3+ arrays or matrices, there are multiple options. First, one could perform multiple two.b.pls analyses and use compare.pls to ascertain which blocks are more "integrated". Second, one could use integration.test and perform a test that averages the amount of integration (correlations) across multiple pairwise blocks. Note that integration.test performed on two matrices or arrays returns the same results as two.b.pls. (Thus, integration.test is more flexible and thorough.)

Using phylogenies and PGLS:

If one wishes to incorporate a phylogeny, phylo.integration is the function to use. This function is exactly the same as integration.test but allows PGLS estimation of PLS vectors. Because integration.test can be used on two blocks, phylo.integration likewise allows one to perform a phylogenetic two-block PLS analysis.

Notes for geomorph 3.0:

There is a slight change in two.b.pls plots with geomorph 3.0. Rather than use the shapes of specimens that matched minimum and maximum PLS scores, major-axis regression is used and the extreme fitted values are used to generate deformation grids. This ensures that shape deformations are exactly along the major axis of shape covariation. This axis is also shown as a best-fit line in the plot.

Notes for geomorph 3.0.4 and subsequent versions:

Compared to previous versions of geomorph, users might notice differences in effect sizes. Previous versions used z-scores calculated with expected values of statistics from null hypotheses (sensu Collyer et al. 2015); however Adams and Collyer (2016) showed that expected values for some statistics can vary with sample size and variable number, and recommended finding the expected value, empirically, as the mean from the set of random outcomes. Geomorph 3.0.4 and subsequent versions now center z-scores on their empirically estimated expected values and where appropriate, log-transform values to assure statistics are normally distributed. This can result in negative effect sizes, when statistics are smaller than expected compared to the avergae random outcome. For ANOVA-based functions, the option to choose among different statistics to measure effect size is now a function argument.

Value

Object of class "pls" that returns a list of the following:

`r.pls`	The correlation coefficient between scores of projected values on the first singular vectors of left (x) and right (y) blocks of landmarks (or other variables). This value can only be negative if single variables are input, as it reduces to the Pearson correlation coefficient.
`P.value`	The empirically calculated P-value from the resampling procedure.
`left.pls.vectors`	The singular vectors of the left (x) block
`right.pls.vectors`	The singular vectors of the right (y) block
`random.r`	The correlation coefficients found in each random permutation of the resampling procedure.
`XScores`	Values of left (x) block projected onto singular vectors.
`YScores`	Values of right (y) block projected onto singular vectors.
`svd`	The singular value decomposition of the cross-covariances. See `svd` for further details.
`A1`	Input values for the left block.
`A2`	Input values for the right block.
`A1.matrix`	Left block (matrix) found from A1.
`A2.matrix`	Right block (matrix) found from A2.
`permutations`	The number of random permutations used in the resampling procedure.
`call`	The match call.

Author(s)

Dean Adams and Michael Collyer

References

Rohlf, F.J., and M. Corti. 2000. The use of partial least-squares to study covariation in shape. Systematic Biology 49: 740-753.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. Heredity. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. Evolution. 70:2623-2631.

See Also

integration.test, modularity.test, phylo.pls, phylo.integration, and compare.pls

Examples

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

#2B-PLS between head shape and food use data
PLS <-two.b.pls(Y.gpa$coords,plethShapeFood$food,iter=999)
summary(PLS)
plot(PLS)
```

---

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 two-dimensional matrix

Usage

```
two.d.array(A, sep = ".")
```

Arguments

A	A 3D array (p x k x n) containing landmark coordinates for a set of specimens
sep	An optional argument for variable labeling, combining landmark labels (e.g., 1, 2, 3, ...) and partial dimension labels (e.g., "x", "y", and "z"), much like the paste function. The default is sep = ".", but this can be changed to any separator. One should make sure to match separators with arrayspecs if switching between matrices and arrays.

Details

This function converts a (p x k x n) array of landmark coordinates into a two-dimensional matrix (n x [p x k]). 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.

Value

Function returns a two-dimensional matrix of dimension (n x [p x k]), where rows represent specimens and columns represent variables.

Author(s)

Dean Adams and Emma Sherratt

See Also

arrayspecs

Examples

```
data(plethodon)
plethodon$land    #original data in the form of 3D array

two.d.array(plethodon$land)   # Convert to a 2D data matrix
```

---

warpRefMesh *Creates a mesh3d object warped to the mean shape*

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

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

Arguments

mesh	A mesh3d object (e.g. made by read.ply)
mesh.coord	A p x k matrix of 3D coordinates digitized on the ply file.
ref	A p x 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 `findMeanSpec`
3. Warp this specimen into the mean shape using `warpRefMesh`
4. Use this average mesh where it asks for a mesh= in the analysis functions and visualization functions

Users should ensure that their mesh and mesh.coord matrix are in the same scale (a common issue is that the mesh is in micrometers and coordinates are in mm or cm). Use range(mesh$vb[1:3,]) and range(mesh.coord) to check and adjust mesh.coord as necessary.

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 (`plotTangentSpace`, `two.b.pls`, `bilat.symmetry`, and `plotRefToTarget`).

Value

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

Author(s)

Emma Sherratt

References

Bookstein, F. L. 1989 Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. IEEE Transactions on Pattern Analysis and Machine Intelligence 11(6):567-585.

Rohlf, F. J. 1998. On Applications of Geometric Morphometrics to Studies of Ontogeny and Phylogeny. Systematic Biology. 47:147-158.

See Also

`findMeanSpec`

`warpRefOutline` *Creates a 2D outline warped to the mean shape*

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.

Usage

```
warpRefOutline(file, coord, ref)
```

Arguments

`file`	A .txt or .csv file of the outline point coordinates, or a .TPS file with OUTLINES= 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`, `two.b.pls`, `bilat.symmetry`, and `plotRefToTarget`).

Value

Function returns an outline object

Author(s)

Emma Sherratt

References

Bookstein, F. L. 1989 Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. IEEE Transactions on Pattern Analysis and Machine Intelligence 11(6):567-585.

Rohlf, F. J. 1998. On Applications of Geometric Morphometrics to Studies of Ontogeny and Phylogeny. Systematic Biology. 47:147-158.

See Also

findMeanSpec

---

writeland.tps *Write landmark data to tps file*

Description

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

Usage

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

Arguments

A	A 3D array (p x k x n) containing landmark coordinates for a set of specimens
file	Name of the *.tps file to be created
scale	A logical value stating whether warnings should be printedAn optional vector containing the length of the scale for each specimen
specID	A logical value stating whether specimen ID names should be saved to line ID=

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