

# Package ‘Arothron’

February 1, 2023

**Type** Package

**Title** Geometric Morphometric Methods and Virtual Anthropology Tools

**Version** 2.0.5

**Author** Antonio Profico, Costantino Buzi, Silvia Castiglione, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**Maintainer** Antonio Profico <antonio.profico@gmail.com>

**Description** Tools for geometric morphometric analysis. The package includes tools of virtual anthropology to align two not articulated parts belonging to the same specimen, to build virtual cavities as endocast (Profico et al, 2021 <[doi:10.1002/ajpa.24340](https://doi.org/10.1002/ajpa.24340)>).

**Depends** R (>= 3.5.0)

**Imports** abind (>= 1.4), alphashape3d (>= 1.3), compositions (>= 1.40-1), doParallel (>= 1.0.11), foreach (>= 1.4.4), geometry (>= 0.3-6), graphics (>= 3.4.0), grDevices (>= 3.4.0), methods (>= 3.5), Morpho (>= 2.5.0), parallel (>= 1.0), rgl (>= 1.0.1), Rvcg (>= 0.17), stats (>= 3.4.0), stats4 (>= 4.0), stringr (>= 1.3.0), utils (>= 3.4.0), vegan (>= 2.4)

**License** GPL-2

**Encoding** UTF-8

**LazyLoad** yes

**RoxygenNote** 7.2.3

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2023-02-01 12:40:08 UTC

## R topics documented:

Arothron-package . . . . .	3
Altapic . . . . .	3
aro.clo.points . . . . .	4
arraytolist . . . . .	4
bary.mesh . . . . .	5

compare_check.set . . . . .	6
CScorrection . . . . .	7
dec.curve . . . . .	8
DM_base_sur . . . . .	9
DM_face_sur . . . . .	9
DM_set . . . . .	10
dta . . . . .	10
endomaker . . . . .	12
endomaker_dir . . . . .	14
endo_set . . . . .	16
export_amira . . . . .	16
export_amira.path . . . . .	17
ext.int.mesh . . . . .	17
ext.mesh.raii . . . . .	19
femsets . . . . .	19
grid_pov . . . . .	20
human_skull . . . . .	20
image2palettes . . . . .	21
krd1_tooth . . . . .	22
landmark_frm2amira . . . . .	22
listtoarray . . . . .	23
localmeshdiff . . . . .	23
Lset2D_list . . . . .	25
Lset3D_array . . . . .	25
malleus_bone . . . . .	26
MAs_sets . . . . .	26
MLESCorrection . . . . .	26
noise.mesh . . . . .	27
out.inn.mesh . . . . .	28
patches_frm2amira . . . . .	29
PCscoresCorr . . . . .	30
permutangle . . . . .	30
pov_selector . . . . .	32
primendoR . . . . .	33
read.amira.dir . . . . .	33
read.amira.set . . . . .	34
read.path.amira . . . . .	34
repmat . . . . .	35
RMs_sets . . . . .	35
SCPI.mesh . . . . .	36
sinus_set . . . . .	36
SM_set . . . . .	36
spherical.flipping . . . . .	37
trasf.mesh . . . . .	37
twodvarshape . . . . .	38
twodviews . . . . .	39
volendo . . . . .	40
yoda_set . . . . .	41

<i>Arothron-package</i>	3
yoda_sur . . . . .	41
<b>Index</b>	<b>42</b>

---

<i>Arothron-package</i>	<i>eometric Morphometric Methods and Virtual Anthropology Tools</i>
-------------------------	---

---

**Description**

Tools for geometric morphometric analysis. The package includes tools of virtual anthropology to align two not articulated parts belonging to the same specimen, to build virtual cavities as endocast (Profico et al, 2021 <doi:10.1002/ajpa.24340>).

**Author(s)**

Antonio Profico, Costantino Buzi, Silvia Castiglione, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

<i>Altapic</i>	<i>example dataset</i>
----------------	------------------------

---

**Description**

2D image of the Altamura man fossil

**Usage**

`data(Altapic)`

**Author(s)**

Antonio Profico

aro.clo.points            *aro.clo.points*

---

**Description**

Find the closest matches between a reference (2D or 3D matrix) and a target matrix (2D/3D) or mesh returning row indices and distances

**Usage**

```
aro.clo.points(target, reference)
```

**Arguments**

target	kxm matrix or object of class "mesh3d"
reference	numeric: a kxm matrix (coordinates)

**Value**

position numeric: a vector of the row indices  
distances numeric: a vector of the coordinates distances

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**Examples**

```
#load an example: mesh, and L set
data(yoda_sur)
data(yoda_set)
sur<-yoda_sur
set<-yoda_set
ver_pos<-aro.clo.points(target=sur,reference=set)
```

---

arraytolist            *arraytolist*

---

**Description**

converts an array in a list storing each element of the third dimension of the array (specimen) as element of the list

**Usage**

```
arraytolist(array)
```

**Arguments**

array            a kx3xn array with landmark coordinates

**Value**

a list containing the landmark configurations stored as separated elements

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

*bary.mesh*

*bary.mesh*

---

**Description**

This function calculates the barycenter of a matrix or a 3D mesh

**Usage**

```
bary.mesh(mesh)
```

**Arguments**

mesh            matrix mesh vertex

**Value**

barycenter numeric: x,y,z coordinates of the barycenter of the mesh

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**Examples**

```
#load an example: mesh, and L set
data(SCP1.mesh)
sur<-SCP1.mesh
bary<-bary.mesh(mesh=sur)
```

---

compare\_check.set      *compare\_check.set*

---

### Description

This function applies the Digital Alignment Tool (DTA) on a disarticulated model using a reference landmark configuration

### Usage

```
compare_check.set(RM_set_1, RM_set_2, DM_set_1, DM_set_2, DM_mesh_1, DM_mesh_2)
```

### Arguments

RM_set_1	matrix: 3D landmark set of the first module acquired on the reference model
RM_set_2	matrix: 3D landmark set of the second module acquired on the reference model
DM_set_1	matrix: 3D landmark set of the first module acquired on the disarticulated model
DM_set_2	matrix: 3D landmark set of the second module acquired on the disarticulated model
DM_mesh_1	mesh3d: mesh of the disarticulated model (first module)
DM_mesh_2	mesh3d: mesh of the disarticulated model (second module)

### Value

SF1 numeric: scale factor used to scale the reference set (first module)  
 SF2 numeric: scale factor used to scale the reference set (second module)  
 RM\_set\_1\_sc matrix: scaled 3D reference set (first module)  
 RM\_set\_2\_sc matrix: scaled 3D reference set (second module)  
 AM\_model list: output of the Morpho::rotmesh.onto function  
 dist\_from\_mesh numeric: mesh distance between the aligned model and the scaled reference set  
 eucl\_dist\_1 numeric: euclidean distance between the landmark configuration of the disarticulated and reference model (first module)  
 eucl\_dist\_2 numeric: euclidean distance between the landmark configuration of the disarticulated and reference model (second module)  
 procr\_dist numeric: procrustes distance between the landmark configuration of the aligned and reference model  
 procr\_dist\_1 numeric: procrustes distance between the landmark configuration of the disarticulated and reference model (first module)  
 procr\_dist\_2 numeric: procrustes distance between the landmark configuration of the disarticulated and reference model (second module)  
 eucl\_dist numeric: euclidean distance between the landmark configuration of the aligned and reference model

single\_1\_1 numeric: euclidean distance between the landmark configuration of the disarticulated and reference model (first module)

single\_1\_2 numeric: euclidean distance between the landmark configuration of the disarticulated and reference model (second module)

### Author(s)

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

CScoreffect	<i>CScoreffect Plot showing the correlation in the shape space between original and combined dataset omitting or including the normalization factors calculated with Arothron and MLECScorection</i>
-------------	--

---

### Description

CScoreffect Plot showing the correlation in the shape space between original and combined dataset omitting or including the normalization factors calculated with Arothron and MLECScorection

### Usage

```
CScoreffect(
  array1,
  array2,
  nPCs = c(1:3),
  from = 0.02,
  to = 0.9,
  length.out = 100
)
```

### Arguments

array1	array: first set of landmark configuration
array2	array: second set of landmark configuration
nPCs	numeric vector: specify which PC scores will be selected in the correlation test
from	numeric: the lower interval of the normalization factor distribution
to	numeric: the lower interval of the normalization factor distribution
length.out	numeric: number of values ranged between from and to

### Value

PCscores PCscores matrix of the combined dataset applying the normalization factor calculated by using the maximum likelihood estimation

PCs PCs matrix of the combined dataset applying the normalization factor calculated by using the maximum likelihood estimation

corr mean correlation between original and combined dataset

CSratios normalization factor calculated by using the maximum likelihood estimation

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**Examples**

```
## Not run:
# Femora case study
data(femsets)
all_pois<-matrix(1:(200*61),nrow=61,ncol=200,byrow = FALSE)
set_ext_100<-femsets[all_pois[,1:100],,]

set_int_100<-femsets[all_pois[,101:200],,]
set_int_50<-set_int_100[c(matrix(1:6100,ncol=61)[seq(1,100,2),]),,]
set_int_20<-set_int_100[c(matrix(1:6100,ncol=61)[seq(1,100,5),]),,]
set.seed(123)
sel<-sample(1:100,10)
set_int_10r<-set_int_100[c(matrix(1:6100,ncol=61)[sel,]),,]

CScoreffect(set_ext_100,set_int_50,nPCs=1:3)
CScoreffect(set_ext_100,set_int_20,nPCs=1:3)
CScoreffect(set_ext_100,set_int_10r,nPCs=1:3)

## End(Not run)
```

---

dec.curve

*dec.curve*

---

**Description**

This function computes the order of points on a open 3D curve and finds intermediate points

**Usage**

```
dec.curve(mat_input, mag, plot = TRUE)
```

**Arguments**

mat_input	numeric: a kx3 matrix
mag	numeric: how many times will be divided by the number of initial points
plot	logical: if TRUE will be plotted the starting and final point matrices

**Value**

matt numeric: a kx3 matrix with points coordinates

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano



**Examples**

```
## Not run:  
## Create and plot a 3D curve  
require(compositions)  
require(rgl)  
curve_3D<-cbind(1:10,seq(1,5,length=10),rnorm(10,sd = 0.2))  
plot3D(curve_3D,bbox=FALSE)  
close3d()  
## Create and plot the new 3D curve (with intermediate points)  
dec_curve_3D<-dec.curve(curve_3D, 2, plot = TRUE)  
  
## End(Not run)
```

---

DM_base_sur	<i>example dataset</i>
-------------	------------------------

---

**Description**

3D mesh of the first part of the Homo sapiens disarticulated model

**Usage**

```
data(DM_base_sur)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

DM_face_sur	<i>example dataset</i>
-------------	------------------------

---

**Description**

3D mesh of the second part of the Homo sapiens disarticulated model

**Usage**

```
data(DM_face_sur)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

DM_set	<i>example dataset</i>
--------	------------------------

---

**Description**

Landmark configurations of the two part of the disarticulated model

**Usage**

```
data(DM_set)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

dta	<i>dta</i>
-----	------------

---

**Description**

This function applies the Digital Alignment Tool (DTA) on a disarticulated model using a reference sample

**Usage**

```
dta(
  RM_sample,
  mod_1,
  mod_2,
  pairs_1,
  pairs_2,
  DM_mesh_1,
  DM_mesh_2,
  DM_set_1,
  DM_set_2,
  method = c("euclidean")
)
```

**Arguments**

RM_sample	3D array: 3D landmark configurations of the reference sample
mod_1	numeric vector: vector containing the position of which landmarks belong to the first module
mod_2	numeric vector: vector containing the position of which landmarks belong to the second module

pairs_1	matrix: a X x 2 matrix containing the indices of right and left landmarks of the first module
pairs_2	matrix: a X x 2 matrix containing the indices of right and left landmarks of the second module
DM_mesh_1	mesh3d: mesh of the disarticulated model (first module)
DM_mesh_2	mesh3d: mesh of the disarticulated model (second module)
DM_set_1	matrix: 3D landmark set of the first module acquired on the disarticulated model
DM_set_2	matrix: 3D landmark set of the second module acquired on the disarticulated model
method	character: specify method to be used to individuate the best DTA ("euclidean" or "procrustes")

### Value

AM\_mesh mesh3d: mesh of the aligned model  
 AM\_set matrix: landmark configuration of the aligned model  
 AM\_id character: name of the item of the reference sample resulted as best DTA  
 AM\_SF\_1 numeric: scale factor used to scale the reference set (first module)  
 AM\_SF\_2 numeric: scale factor used to scale the reference set (second module)  
 distance numeric: distance between the landmark configuration of the aligned and the reference model  
 tot\_proc numeric vector: procrustes distances between aligned and reference models (all DTAs)  
 tot\_eucl numeric vector: euclidean distances between aligned and reference models (all DTAs)  
 setarray 3D array: landmark configurations of the disarticulated model aligned on each item of the reference sample

### Author(s)

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

### References

Profico, A., Buzi, C., Davis, C., Melchionna, M., Veneziano, A., Raia, P., & Manzi, G. (2019). A new tool for digital alignment in Virtual Anthropology. *The Anatomical Record*, 302(7), 1104-1115.

### Examples

```
## Load and plot the disarticulated model of the Homo sapiens case study
library(compositions)
library(rgl)
data(DM_base_sur)
data(DM_face_sur)
open3d()
wire3d(DM_base_sur,col="white")
wire3d(DM_face_sur,col="white")
```

```

## Load the landmark configurations associated to the DM
data(DM_set)
## Load the reference sample
data(RMs_sets)
## Define the landmarks belonging to the first and second module
mod_1<-c(1:17) #cranial base
mod_2<-c(18:32) #facial complex
## Define the paired landmarks for each module (optional symmetrization process)
pairs_1<-cbind(c(4,6,8,10,12,14,16),c(5,7,9,11,13,15,17))
pairs_2<-cbind(c(23,25,27,29,31),c(24,26,28,30,32))
## Run DTA
ex.dta<-dta(RM_sample=RMs_sets, mod_1=mod_1, mod_2=mod_2, pairs_1=pairs_1, pairs_2=pairs_2,
DM_mesh_1=DM_base_sur,DM_mesh_2=DM_face_sur, DM_set_1= DM_set[mod_1,], DM_set_2=DM_set[mod_2,])
## Print the name of the best RM
ex.dta$AM_id
## Save the mesh and the landmark set of the AM
AM_mesh<-ex.dta$AM_mesh
AM_set<-ex.dta$AM_set
## Plot the aligned 3D model
library(compositions)
library(rgl)
open3d()
wire3d(AM_mesh,col="white")
plot3D(AM_set,bbox=FALSE,add=TRUE)

```

---

endomaker

*endomaker*

---

## Description

Build endocast from a skull 3D mesh

## Usage

```

endomaker(
  mesh = NULL,
  path_in = NULL,
  param1_endo = 1,
  npovs = 50,
  volume = TRUE,
  alpha_vol = 100,
  nVoxels = 1e+05,
  decmesh = 20000,
  alpha_ext = 30,
  ncells = 50000,
  npovs_calse = 50,
  param1_calse = 2,
  param1_ast = 1.3,
  decendo = 20000,

```

```

    scalendo = 0.5,
    alpha_end = 100,
    mpovdist = 10,
    plot = FALSE,
    colmesh = "orange",
    save = FALSE,
    outpath = tempdir(),
    num.cores = NULL
)

```

### Arguments

mesh	mesh3d: 3D model of the skull
path_in	character: path of the skull where is stored
param1_endo	numeric: parameter for spherical flipping
npovs	numeric: number of Points of View used in the endocast construction
volume	logical: if TRUE the calculation of the volume (expressed in cc) through concave is returned
alpha_vol	numeric: alpha shape for volume calculation
nVoxels	numeric: number of voxels for estimation endocranial volume
decmesh	numeric: decmesh
alpha_ext	numeric: alpha shape for construction external cranial mesh
ncells	numeric: approximative number of cell for 3D grid construction
npovs_calse	numeric: number of Points of View for construction of skull shell
param1_calse	numeric: parameter for calse (construction shell)
param1_ast	numeric: parameter for ast3d (construction row endocast)
decendo	numeric: desired number of triangles (row endocast)
scalendo	numeric: scale factor row endocast (for definition of POVs)
alpha_end	numeric: alpha shape value for concave hull (row endocast)
mpovdist	numeric: mean value between POVs and mesh
plot	logical: if TRUE the endocast is plotted
colmesh	character: color of the mesh to be plotted
save	logical: if TRUE the mesh of the endocast is saved
outpath	character: path where save the endocast
num.cores	numeric: numbers of cores to be used in parallel elaboration

### Value

endocast mesh3d: mesh of the endocast  
 volume numeric: volume of the endocast expressed in cc

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**References**

Profico, A., Buzi, C., Melchionna, M., Veneziano, A., & Raia, P. (2020). Endomaker, a new algorithm for fully automatic extraction of cranial endocasts and the calculation of their volumes. *American Journal of Physical Anthropology*.

**Examples**

```
## Not run:
library(rgl)
data(human_skull)
sapendo<-endomaker(human_skull,param1_endo = 1.0,decmesh = 20000, num.cores=NULL)
open3d()
wire3d(sapendo$endocast,col="violet")
ecv<-sapendo$volume

## End(Not run)
```

---

endomaker\_dir

*endomaker\_dir*

---

**Description**

Build library of endocasts from skull 3D meshes

**Usage**

```
endomaker_dir(
  dir_path,
  param1_endo = 1.5,
  npovs = 50,
  volume = TRUE,
  alpha_vol = 50,
  nVoxels = 1e+05,
  decmesh = 20000,
  alpha_ext = 30,
  ncells = 50000,
  npovs_calse = 50,
  param1_calse = 3,
  param1_ast = 1.3,
  decendo = 20000,
  scalendo = 0.5,
  alpha_end = 100,
  mpovdist = 10,
  plotall = FALSE,
```

```

    colmesh = "orange",
    save = FALSE,
    outpath = tempdir(),
    num.cores = NULL
)

```

### Arguments

dir_path	character: path of the folder where the skull meshes are stored
param1_endo	numeric vector: parameter for spherical flipping
npovs	numeric: number of Points of View used in the endocast construction
volume	logical: if TRUE the volume of the endocast (ECV) is estimated
alpha_vol	numeric: alpha shape for volume calculation
nVoxels	numeric: number of voxels for estimation endocranial volume
decmesh	numeric: decmesh
alpha_ext	numeric: alpha shape for construction external cranial mesh
ncells	numeric: approximative number of cell for 3D grid construction
npovs_calse	numeric: number of Points of View for construction of skull shell
param1_calse	numeric: parameter for calse (construction shell)
param1_ast	numeric: parameter for ast3d (construction row endocast)
decendo	numeric: desired number of triangles (row endocast)
scalendo	numeric: scale factor row endocast (for definition of POVs)
alpha_end	numeric: alpha shape value for concave hull (row endocast)
mpovdist	numeric vector: mean value between POVs and mesh
plotall	logical: if TRUE the endocasts are plotted
colmesh	character: color of the mesh to be plotted
save	logical: if TRUE the mesh of the endocast is saved
outpath	character: path where save the endocast
num.cores	numeric: number of cores to be used in parallel elaboration

### Value

endocasts mesh3d: list of meshes of the extracted endocasts  
 volumes numeric: volumes of the endocasts expressed in cc

### Author(s)

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

### References

Profico, A., Buzi, C., Melchionna, M., Veneziano, A., & Raia, P. (2020). Endomaker, a new algorithm for fully automatic extraction of cranial endocasts and the calculation of their volumes. *American Journal of Physical Anthropology*.

---

endo_set	<i>example dataset</i>
----------	------------------------

---

**Description**

POVs defined inside the endocranial cavity

**Usage**

```
data(endo_set)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

export_amira	<i>export_amira</i>
--------------	---------------------

---

**Description**

This function exports a list of 3D landmark set in separate files (format landmarkAscii)

**Usage**

```
export_amira(lista, path)
```

**Arguments**

lista	list containing 3D landmark sets
path	character: path of the folder where saving the Amira landmark sets

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**Examples**

```
x<-c(1:20)
y<-seq(1,3,length=20)
z<-rnorm(20,0.01)
vertices<-cbind(x,y,z)
set<-list(vertices)
example<-export_amira(set,path=tempdir())
```



---

export_amira.path	<i>export_amira.path</i>
-------------------	--------------------------

---

### Description

Convert and save a 3D matrix into a AmiraMesh ASCII Lineset (.am) object

### Usage

```
export_amira.path(  
  vertices,  
  filename,  
  Lines = c(1:(dim(vertices)[1] - 1) - 1, -1),  
  path  
)
```

### Arguments

vertices	numeric: a kx3 matrix
filename	character: name of the requested output
Lines	numeric: sequence of the vertices that defines the line
path	character: folder path

### Author(s)

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

### Examples

```
x<-c(1:20)  
y<-seq(1,3,length=20)  
z<-rnorm(20,0.01)  
vertices<-cbind(x,y,z)  
export_amira.path(vertices=vertices,filename="example_line",path=tempdir())
```

---

ext.int.mesh	<i>ext.int.mesh</i>
--------------	---------------------

---

### Description

This function finds the vertices visible from a set of points of view

**Usage**

```

ext.int.mesh(
  mesh,
  views = 20,
  dist.sphere = 3,
  param1 = 2.5,
  param2 = 10,
  default = TRUE,
  import_pov,
  matrix_pov,
  expand = 1,
  scale.factor,
  method = "ast3d",
  start.points = 250,
  num.cores = NULL
)

```

**Arguments**

mesh	object of class mesh3d
views	numeric: number of points of view
dist.sphere	numeric: scale factor. This parameter the distance between the barycenter of the mesh and the radius of the sphere used to define set of points of view
param1	numeric: first parameter for spherical flipping (usually ranged from 0.5 to 5, try!)
param2	numeric second paramter for spherical flipping (don't change it!)
default	logical: if TRUE the points of views are defined automatically, if FALSE define the matrix_pov
import_pov	logical: if NULL an interactive 3D plot for the definition of the points of view is returned
matrix_pov	matrix: external set of points of view
expand	numeric: scale factor for the grid for the interactive 3D plot
scale.factor	numeric: scale factor for sphere inscribed into the mesh
method	character: select "a" or "c"
start.points	numeric: number of POVs available
num.cores	numeric: number of cores

**Value**

position numeric: a vector with vertex number nearest the landmark set

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**References**

Profico, A., Schlager, S., Valoriani, V., Buzi, C., Melchionna, M., Veneziano, A., ... & Manzi, G. (2018). Reproducing the internal and external anatomy of fossil bones: Two new automatic digital tools. *American Journal of Physical Anthropology*, 166(4), 979-986.

---

`ext.mesh.raï`*ext.mesh.raï*

---

**Description**

This function returns a 3D mesh with colours based on the vertices visible from each point of view

**Usage**

```
ext.mesh.raï(scans, mesh)
```

**Arguments**

scans	an ext.int.mesh
mesh	matrix mesh vertex (the same of the ext.int.mesh object)

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

`femsets`*example dataset*

---

**Description**

3D semilandmark configurations of 21 human femora

**Usage**

```
data(femsets)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

grid_pov	<i>grid_pov</i>
----------	-----------------

---

**Description**

This function creates a grid for an interactive way to define the set of the points of view

**Usage**

```
grid_pov(mesh, expand = 1)
```

**Arguments**

mesh	object of class mesh3d
expand	numeric: scale factor for the grid for the interactive 3D plot

**Value**

matrice matrix: matrix with the x,y,z coordinates of the points of view

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

human_skull	<i>example dataset</i>
-------------	------------------------

---

**Description**

3D mesh of a human skull

**Usage**

```
data(human_skull)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

image2palettes      *image2palettes*

---

### Description

Create palettes from an image

### Usage

```
image2palettes(  
  array,  
  resize = 4,  
  unique = FALSE,  
  scale = F,  
  k = 3,  
  lcols = 7,  
  plsaxis = 1,  
  cex = 5,  
  cext = 0.5  
)
```

### Arguments

array	array: rgb array
resize	numeric: desired resize factor
unique	logical: if TRUE each color is counted once
scale	logical: if TRUE (color) variables are scaled
k	numeric: desired number of clusters (i.e., number of palettes)
lcols	numeric: length of the color vector of each palette
plsaxis	numeric: desired PLS axis
cex	numeric: size of colored squares
cext	numeric: size of color names

### Value

paletteslist list: color palettes arranged in a list

### Author(s)

Antonio Profico

**Examples**

```
## Not run:
require(jpeg)
require(Morpho)
data("Altapic")
image2palettes(Altapic,resize=1,unique=T,scale=T,k=3,lcols=5,plsaxis=1,cext=0.5)

## End(Not run)
```

---

krd1_tooth	<i>example dataset</i>
------------	------------------------

---

**Description**

3D mesh of a deciduous Neanderthal tooth

**Usage**

```
data(krd1_tooth)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

landmark_frm2amira	<i>landmark_frm2amira</i>
--------------------	---------------------------

---

**Description**

This function converts the .frm files, from Evan Toolbox, stored in a folder into the format landmarkAscii

**Usage**

```
landmark_frm2amira(path_folder_frm, path_amira_folder)
```

**Arguments**

path\_folder\_frm  
character: path of the folder where the .frm files are stored

path\_amira\_folder  
character: path folder to store the landmarkAscii configurations

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

listtoarray	<i>listtoarray convert a list into an array</i>
-------------	---

---

**Description**

listtoarray convert a list into an array

**Usage**

```
listtoarray(mylist)
```

**Arguments**

mylist            a list

**Value**

a kx3xn array with landmark coordinates

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

localmeshdiff	<i>localmeshdiff Calculate and Visualize local differences between two meshes</i>
---------------	---

---

**Description**

localmeshdiff Calculate and Visualize local differences between two meshes

**Usage**

```
localmeshdiff(  
  mesh1,  
  mesh2,  
  ploton = 1,  
  diffarea = ((area_shape1 - area_shape2)/area_shape2) * 100,  
  paltot = rainbow(200),  
  from = NULL,  
  to = NULL,  
  n.int = 200,  
  out.rem = TRUE,  
  fact = 1.5,  
  visual = 1,  
  scale01 = TRUE,  
  colwire = "pink"  
)
```

**Arguments**

mesh1	reference mesh: object of class "mesh3d"
mesh2	target mesh: object of class "mesh3d"
ploton	numeric: define which mesh will be used to visualize local differences
diffarea	formula: define how calculating differences in area. area_shape1 refers to mesh1, area_shape2 refers to mesh2
paltot	character vector: specify the colors which are used to create a color palette
from	numeric: minimum distance to be colorised
to	numeric: maximum distance to be colorised
n.int	numeric: determines break points for color palette
out.rem	logical: if TRUE outliers will be removed
fact	numeric: factor k of the interquartile range
visual	numeric: if equals to 1 the mesh is plotted without a wireframe, if set on 2 a wireframe is added
scale01	logical: if TRUE the vector of distances is scaled from 0 to 1
colwire	character: color of the wireframe

**Value**

vect numeric vector containing local differences in area between the reference and the target mesh

**Author(s)**

Antonio Profico, Costantino Buzi, Silvia Castiglione, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**References**

Melchionna, M., Profico, A., Castiglione, S., Sansalone, G., Serio, C., Mondanaro, A., ... & Manzi, G. (2020). From smart apes to human brain boxes. A uniquely derived brain shape in late hominins clade. *Frontiers in Earth Science*, 8, 273.

**Examples**

```
## Not run:
library(Arothron)
library(rgl)
data("primendoR")
neaset<-primendoR$sets[, ,11]
sapset<-primendoR$sets[, ,14]
#defining a mesh for the neanderthal right hemisphere
neasur<-list("vb"=t(cbind(neaset,1)), "it"=primendoR$sur$it)
class(neasur)<-"mesh3d"
#defining a mesh for the modern human right hemisphere
sapsur<-list("vb"=t(cbind(sapset,1)), "it"=primendoR$sur$it)
class(neasur)<-"mesh3d"
```



```
layout3d(t(c(1,2)),sharedMouse = TRUE)
localmeshdiff(sapsur,neasure,1,scale01 = TRUE,
paltot=c("darkred","red","orange","white","lightblue","blue","darkblue"))
next3d()
localmeshdiff(neasure,sapsur,1,scale01 = TRUE,
paltot=c("darkred","red","orange","white","lightblue","blue","darkblue"))

## End(Not run)
```

---

Lset2D_list	<i>example dataset</i>
-------------	------------------------

---

**Description**

List containing five 2D-landmark configurations acquired along five different anatomical views

**Usage**

```
data(Lset2D_list)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

Lset3D_array	<i>example dataset</i>
--------------	------------------------

---

**Description**

Array containing a cranial 3D-landmark configuration acquired on a Primate sample

**Usage**

```
data(Lset3D_array)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

malleus_bone	<i>example dataset</i>
--------------	------------------------

---

**Description**

3D mesh of a human malleus

**Usage**

data(malleus\_bone)

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

MAs_sets	<i>example dataset</i>
----------	------------------------

---

**Description**

Landmark configurations of the manual alignments

**Usage**

data(MAs\_sets)

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

MLECScorection	<i>MLECScorection Maximum Likelihood Estimation of the normalization factor to be applied to optimize the correlation between two landmark configurations to be combined by using twodviews and</i>
----------------	---

---

**Description**

MLECScorection Maximum Likelihood Estimation of the normalization factor to be applied to optimize the correlation between two landmark configurations to be combined by using twodviews and

**Usage**

MLECScorection(array1, array2, scale = TRUE, nPCs = 1:5)

**Arguments**

array1	array: first set of landmark configuration
array2	array: second set of landmark configuration
scale	logical: if FALSE the analysis is performed in the shape space, if TRUE the analysis is performed in the size and shape space (gpa without scaling)
nPCs	numeric vector: specify which PC scores will be selected in the correlation test

**Value**

PCscores PCscores matrix of the combined dataset applying the normalization factor calculated by using the maximum likelihood estimation

PCs PCs matrix of the combined dataset applying the normalization factor calculated by using the maximum likelihood estimation

corr mean correlation between original and combined dataset

CSratios normalization factor calculated by using the maximum likelihood estimation

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

noise.mesh	<i>noise.mesh</i>
------------	-------------------

---

**Description**

This function adds noise to a mesh

**Usage**

```
noise.mesh(mesh, noise = 0.025, seed = 123)
```

**Arguments**

mesh	triangular mesh stored as object of class "mesh3d"
noise	sd deviation to define vertex noise
seed	seed for random number generator

**Value**

mesh\_n a 3D model of class "mesh3d" with noise

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**Examples**

```
#load mesh
library(compositions)
library(rgl)
data("SCP1.mesh")
mesh<-SCP1.mesh
#add noise
noised<-noise.mesh(mesh,noise=0.05)
#plot original and mesh with noise added
open3d()
shade3d(mesh,col=3)
shade3d(noised,col=2,add=TRUE)
```

---

out.inn.mesh

*out.inn.mesh*


---

**Description**

This function separates a 3D mesh subjected to the ext.int.mesh into two 3D models: the visible mesh and the not visible one

**Usage**

```
out.inn.mesh(scans, mesh, plot = TRUE)
```

**Arguments**

scans	an ext.int.mesh
mesh	matrix mesh vertex (the same of the ext.int.mesh object)
plot	logical: if TRUE the wireframe of the mesh with the visible vertices is plotted

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**Examples**

```
## Not run:
#CA-LSE tool on Neanderthal tooth
#load a mesh
data(krd1_tooth)
library(rgl)
library(Rvcg)
library(compositions)
ca_lse_krd1<-ext.int.mesh(mesh= krd1_tooth, views=50, param1=3, default=TRUE,
import_pov = NULL,expand=1, scale.factor=1,num.cores = NULL)
vis_inv_krd1<-out.inn.mesh(ca_lse_krd1, krd1_tooth, plot=TRUE)
inv_mesh<-vcgIsolated(vis_inv_krd1$invisible)
```

```

open3d()
shade3d(inv_mesh,col=2)
open3d()
shade3d(vis_inv_krd1$visible, col=3)
#CA-LSE tool on human malleus
#load a mesh
data(malleus_bone)
ca_lse_malleus<-ext.int.mesh(mesh= malleus_bone, views=50, param1=3,
default=TRUE, import_pov = NULL, expand=1, scale.factor=1)
vis_inv_malleus<-out.inn.mesh(ca_lse_malleus, malleus_bone, plot=TRUE)
inv_mesh<- vis_inv_malleus$invisible
inv_mesh<-ca_lse_malleus$invisible

#AST-3D tool
#load a mesh
data(human_skull)
data(endo_set)
ast3d_endocast<-ext.int.mesh(mesh=human_skull, views=50, param1=0.6, default=FALSE,
import_pov = TRUE,expand=1, matrix_pov =endo_set, scale.factor=1,num.cores = NULL)
vis_inv_endo<-out.inn.mesh(ast3d_endocast,human_skull,plot=TRUE)
vis_mesh<-vcgIsolated(vis_inv_endo$visible)
open3d()
shade3d(vis_mesh,col=3)
open3d()
shade3d(vis_inv_endo$invisible, col=2)

## End(Not run)

```

---

patches\_frm2amira      *patches\_frm2amira*

---

## Description

This function converts the .frm files, from Evan Toolbox, stored in a folder into the format landmarkAscii (semilandmark patches)

## Usage

```
patches_frm2amira(path_folder_frm, path_amira_folder)
```

## Arguments

path\_folder\_frm  
character: path of the folder where the .frm files are stored

path\_amira\_folder  
character: path folder to store the landmarkAscii configurations

## Author(s)

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

PCscoresCorr	<i>PCscoresCorr</i> Perform a correlation test between two matrices of PC-scores
--------------	--

---

**Description**

PCscoresCorr Perform a correlation test between two matrices of PCscores

**Usage**

```
PCscoresCorr(matrix1, matrix2, nPCs = 1:5)
```

**Arguments**

matrix1	matrix: first set of PC scores
matrix2	matrix: second set of PC scores
nPCs	numeric vector: specify which PC scores will be selected in the correlation test

**Value**

corr the correlation values associated to each pair of PC scores  
 p.values p-values associated to the correlation test

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

permutangle	<i>permutangle</i>
-------------	--------------------

---

**Description**

Create palettes from an image

**Usage**

```
permutangle(
  mat,
  var,
  group1,
  group2,
  scale = FALSE,
  iter = 100,
  cex1 = range01(var[group1] + 1),
  cex2 = range01(var[group2] + 1),
```

```

    cex3 = 0.7,
    cex4 = 1.2,
    labels = c("stgr1", "stgr2", "endgr1", "endgr2"),
    pch1 = 19,
    pch2 = 19,
    pch3 = 19,
    col1 = "red",
    col2 = "blue"
  )

```

### Arguments

mat	array: rgb array
var	numeric: desired resize factor
group1	logical: if TRUE each color is counted once
group2	logical: if TRUE (color) variables are scaled
scale	numeric: desired number of clusters (i.e., number of palettes)
iter	numeric: length of the color vector of each palette
cex1	numeric: desired PLS axis
cex2	numeric: size of colored squares
cex3	numeric: size of color names
cex4	numeric: size of color names
labels	numeric: size of color names
pch1	numeric: size of color names
pch2	numeric: size of color names
pch3	numeric: size of color names
col1	numeric: size of color names
col2	numeric: size of color names

### Value

angle list: color palettes arranged in a list  
 permangles list: color palettes arranged in a list  
 angle list: color palettes arranged in a list  
 iterangles list: color palettes arranged in a list  
 p-value list: color palettes arranged in a list  
 PCA\_angle list: color palettes arranged in a list  
 PCA\_interangles list: color palettes arranged in a list  
 PCA\_p-value list: color palettes arranged in a list

### Author(s)

Antonio Profico

**Examples**

```
## Not run:
require(shapes)
require(Morpho)
data("gorf.dat")
data("gorm.dat")
Array<-bindArr(gorf.dat,gorm.dat,along=3)
CS<-apply(Array,3,cSize)
Sex<-c(rep("F",dim(gorf.dat)[3]),rep("M",dim(gorm.dat)[3]))

#Shape and size space
AllTrajFB<-permutangle(procSym(Array,scale=FALSE,CSinit = FALSE)$PCscores,
var=CS,group1=which(Sex=="F"),group2=which(Sex=="M"),scale=FALSE,iter=50)
hist(AllTrajFB$iterangles,breaks = 100,xlim=c(0,90))
abline(v=AllTrajFB$angle,lwd=2,col="red")
hist(AllTrajFB$PCA_iterangles,breaks = 100,xlim=c(0,90))
abline(v=AllTrajFB$PCA_angle,lwd=2,col="red")

#Shape space
AllTrajFB<-permutangle(procSym(Array)$PCscores,
var=CS,group1=which(Sex=="F"),group2=which(Sex=="M"),scale=FALSE,iter=50)
hist(AllTrajFB$iterangles,breaks = 100,xlim=c(0,90))
abline(v=AllTrajFB$angle,lwd=2,col="red")
hist(AllTrajFB$PCA_iterangles,breaks = 100,xlim=c(0,90))
abline(v=AllTrajFB$PCA_angle,lwd=2,col="red")

## End(Not run)
```

---

pov\_selector

*pov\_selector*


---

**Description**

Internal function to define the points of view

**Usage**

```
pov_selector(mesh, grid, start.points = 250, method = "ast3d")
```

**Arguments**

mesh	object of class mesh3d
grid	matrix: a 3D grid
start.points	numeric: number of center to be found
method	character: select "a" or "c" for respectively AST-3D and CA-LSE method

**Value**

selection numeric: positioning vector of the selected points of the grid



**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

primendoR                      *example dataset*

---

**Description**

right brain hemisphere of 19 primate species

**Usage**

```
data(primendoR)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

read.amira.dir                      *read.amira.dir*

---

**Description**

This function reads and stores in an array the coordinated allocated in a folder in separate files (format landmarkAscii)

**Usage**

```
read.amira.dir(path.dir, nland)
```

**Arguments**

path.dir                      character: path of the folder  
nland                          numeric: number of landmark sampled in Amira

**Value**

array.set numeric: a kx3xn array with landmark coordinates

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

<code>read.amira.set</code>	<i>read.amira.set</i>
-----------------------------	-----------------------

---

**Description**

This function converts a landmarkAscii file set in a kx3x1 array

**Usage**

```
read.amira.set(name.file, nland)
```

**Arguments**

<code>name.file</code>	character: path of a landmarkAscii file
<code>nland</code>	numeric: number of landmark sampled in Amira, if is set on "auto" it will be automatically recognized

**Value**

array.set numeric: a kx3x1 array with landmark coordinates

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

<code>read.path.amira</code>	<i>read.path.amira</i>
------------------------------	------------------------

---

**Description**

This function extracts and orders the coordinate matrix from a surface path file from Amira

**Usage**

```
read.path.amira(path.name)
```

**Arguments**

<code>path.name</code>	character: path of surface path .ascii extension file
------------------------	---

**Value**

data numeric: a kxd matrix with xyz coordinates

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

repmat	<i>repmat</i>
--------	---------------

---

**Description**

This function repeats copies of a matrix

**Usage**

```
repmat(X, m, n)
```

**Arguments**

X	numeric: a matrix
m	numeric: number of times to repeat the X matrix in row and column dimension
n	numeric: repetition factor for each dimension

**Value**

matrice: repeated matrix

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

RMs_sets	<i>example dataset</i>
----------	------------------------

---

**Description**

Array containing the landmark coordinates of the reference sample for Digital Alignment Tool example

**Usage**

```
data(RMs_sets)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

SCP1.mesh	<i>example dataset</i>
-----------	------------------------

---

**Description**

Mesh of the Saccopastore 1 Neanderthal skull

**Usage**

```
data(SCP1.mesh)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

sinus_set	<i>example dataset</i>
-----------	------------------------

---

**Description**

POVs sampled inside the maxillary sinus cavity

**Usage**

```
data(sinus_set)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

SM_set	<i>example dataset</i>
--------	------------------------

---

**Description**

Landmark configuration associated to the starting model

**Usage**

```
data(SM_set)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

spherical.flipping      *spherical.flipping*

---

**Description**

Internal spherical flipping function

**Usage**

```
spherical.flipping(C, mesh, param1, param2)
```

**Arguments**

C	numeric: coordinates of the point of view
mesh	object of class mesh3d
param1	numeric: first parameter for spherical flipping (usually ranged from 0.1 to 3, try!)
param2	numeric second paramter for spherical flipping (don't change it!)

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**References**

Profico, A., Schlager, S., Valoriani, V., Buzi, C., Melchionna, M., Veneziano, A., ... & Manzi, G. (2018). Reproducing the internal and external anatomy of fossil bones: Two new automatic digital tools. *American Journal of Physical Anthropology*, 166(4), 979-986.#' @export

Katz, S., Tal, A., & Basri, R. (2007). Direct visibility of point sets. In *ACM SIGGRAPH 2007 papers* (pp. 24-es).

---

trasf.mesh      *trasf.mesh*

---

**Description**

This function centers a mesh on the barycenter coordinates

**Usage**

```
trasf.mesh(mesh, barycenter)
```

**Arguments**

mesh	a 3D mesh of class "mesh3d"
barycenter	numeric: coordinates of the center

**Value**

mesh a 3D mesh of class "mesh3d"

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

twodvarshape	<i>twodvarshape</i> Calculates the shape variation associated to a value of PC scores associated to a specific combined landmark configuration or view
--------------	--

---

**Description**

twodvarshape Calculates the shape variation associated to a value of PC scores associated to a specific combined landmark configuration or view

**Usage**

```
twodvarshape(twodviews_ob, scores, PC, view)
```

**Arguments**

twodviews_ob	object from twodviews()
scores	numeric: the values of the PC scores for which the visualization is called
PC	PC chosen
view	numeric: which landmark configuration will be used to build the shape variation

**Value**

mat matrix of coordinates associated to the called shape variation

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**References**

Profico, A., Piras, P., Buzi, C., Del Bove, A., Melchionna, M., Senczuk, G., ... & Manzi, G. (2019). Seeing the wood through the trees. Combining shape information from different landmark configurations. *Hystrix*, 157-165.

**Examples**

```

library(Arothron)
#load the 2D primate dataset
data("Lset2D_list")
#combine the 2D datasets and PCA
combin2D<-twodviews(Lset2D_list,scale=TRUE,vector=c(1:5))
#calculate the shape variation associated to the negative extreme value of PC1
min_PC1<-twodvarshape(combin2D,min(combin2D$PCscores[,1]),1,5)
plot(min_PC1,asp=1)
#calculate the shape variation associated to the positive extreme value of PC1
max_PC1<-twodvarshape(combin2D,max(combin2D$PCscores[,1]),1,5)
plot(max_PC1,asp=1)

```

---

twodviews	<i>twodviews Combine and calculate the PCscores matrix from a list of different landmark configurations to be combined</i>
-----------	--

---

**Description**

twodviews Combine and calculate the PCscores matrix from a list of different landmark configurations to be combined

**Usage**

```
twodviews(twodlist, scale = TRUE, vector = NULL)
```

**Arguments**

twodlist	a list containing the landmark configurations of each anatomical view stored as separated lists
scale	logical: TRUE for shape-space, FALSE for form-space
vector	numeric vector: defines which views are to be used

**Value**

PCscores PC scores  
 PCs Pricipal Components (eigenvector matrix)  
 Variance table of the explained variance by the PCs  
 size vector containing the Centroid Size of each configuration  
 mshapes a list containing the mean shape of each landmark configuration  
 dims number of landmarks of each configuration  
 dimm dimension (2D or 3D) of each combined landmark configuration  
 twodlist the list used as input

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**References**

Profico, A., Piras, P., Buzi, C., Del Bove, A., Melchionna, M., Senczuk, G., ... & Manzi, G. (2019). Seeing the wood through the trees. Combining shape information from different landmark configurations. *Hystrix*, 157-165.

**Examples**

```
library(Morpho)
#load the 2D primate dataset
data("Lset2D_list")
length(Lset2D_list)
#combine the 2D datasets and PCA
combin2D<-twodviews(Lset2D_list,scale=TRUE,vector=c(1:5))
combin2D$size
#plot of the first two Principal Components
plot(combin2D$PCscores)
text(combin2D$PCscores,labels=rownames(combin2D$PCscores))
#load the 3D primate dataset
data("Lset3D_array")
#GPA and PCA
GPA_3D<-procSym(Lset3D_array)
#plot of the first two Principal Components
plot(GPA_3D$PCscores)
text(GPA_3D$PCscores,labels=rownames(GPA_3D$PCscores))
```

---

volendo

*volendo*

---

**Description**

Calculate the volume of a mesh by using a voxel-based method

**Usage**

```
volendo(mesh, alpha_vol = 100, ncells = 1e+05)
```

**Arguments**

mesh	object of class mesh3d
alpha_vol	numeric: alpha shape for construction external concave hull
ncells	numeric: approximative number of cell for 3D grid construction

**Value**

vol numeric: volume of the mesh expressed in cc



**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

**Examples**

```
## Not run:  
#load the human skull  
library(rgl)  
data(human_skull)  
sapendo<-endomaker(human_skull,param1_endo = 1.0,vol=FALSE, num.cores=NULL)  
volsap<-volendo(sapendo$endocast)  
  
## End(Not run)
```

---

yoda_set	<i>example dataset</i>
----------	------------------------

---

**Description**

Landmark set on Yoda

**Usage**

```
data(yoda_set)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

---

yoda_sur	<i>example dataset</i>
----------	------------------------

---

**Description**

Mesh of Yoda

**Usage**

```
data(yoda_sur)
```

**Author(s)**

Antonio Profico, Costantino Buzi, Marina Melchionna, Paolo Piras, Pasquale Raia, Alessio Veneziano

# Index

## \* Arothron

- Altapic, [3](#)
- DM\_base\_sur, [9](#)
- DM\_face\_sur, [9](#)
- DM\_set, [10](#)
- endo\_set, [16](#)
- femsets, [19](#)
- human\_skull, [20](#)
- krd1\_tooth, [22](#)
- Lset2D\_list, [25](#)
- Lset3D\_array, [25](#)
- malleus\_bone, [26](#)
- MAs\_sets, [26](#)
- primendoR, [33](#)
- RMs\_sets, [35](#)
- SCP1.mesh, [36](#)
- sinus\_set, [36](#)
- SM\_set, [36](#)
- yoda\_set, [41](#)
- yoda\_sur, [41](#)

Altapic, [3](#)

aro.clo.points, [4](#)

Arothron (Arothron-package), [3](#)

Arothron-package, [3](#)

arraytolist, [4](#)

bary.mesh, [5](#)

compare\_check.set, [6](#)

CScorreffect, [7](#)

dec.curve, [8](#)

DM\_base\_sur, [9](#)

DM\_face\_sur, [9](#)

DM\_set, [10](#)

dta, [10](#)

endo\_set, [16](#)

endomaker, [12](#)

endomaker\_dir, [14](#)

export\_amira, [16](#)

export\_amira.path, [17](#)

ext.int.mesh, [17](#)

ext.mesh.raii, [19](#)

femsets, [19](#)

grid\_pov, [20](#)

human\_skull, [20](#)

image2palettes, [21](#)

krd1\_tooth, [22](#)

landmark\_frm2amira, [22](#)

listtoarray, [23](#)

localmeshdiff, [23](#)

Lset2D\_list, [25](#)

Lset3D\_array, [25](#)

malleus\_bone, [26](#)

MAs\_sets, [26](#)

MLECSCorrection, [26](#)

noise.mesh, [27](#)

out.inn.mesh, [28](#)

patches\_frm2amira, [29](#)

PCscoresCorr, [30](#)

permutangle, [30](#)

pov\_selector, [32](#)

primendoR, [33](#)

read.amira.dir, [33](#)

read.amira.set, [34](#)

read.path.amira, [34](#)

repmat, [35](#)

RMs\_sets, [35](#)

SCP1.mesh, [36](#)

sinus\_set, 36

SM\_set, 36

spherical.flipping, 37

trasf.mesh, 37

twodvarshape, 38

twodviews, 39

volendo, 40

yoda\_set, 41

yoda\_sur, 41