

Package ‘Arothron’

July 16, 2018

Type Package

Title Geometric Morphometrics Analyses

Version 1.0.2

Author Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Maintainer Antonio Profico <antonio.profico@uniroma1.it>

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, 2018 <doi:10.1002/ajpa.23493>), and functions to import and export the coordinates of landmarks and 3D paths into 'landmarkAscii' and 'am' format files.

Depends R (>= 3.4.0)

Imports compositions (>= 1.40-1), doParallel (>= 1.0.11), foreach (>= 1.4.4), geometry (>= 0.3-6), graphics (>= 3.4.0), grDevices (>= 3.4.0), Morpho (>= 2.5.0), rgl (>= 0.93.0), Rvcg (>= 0.17), stats (>= 3.4.0), stringr (>= 1.3.0), utils (>= 3.4.0), vegan (>= 2.4)

License GPL-2

Encoding UTF-8

LazyLoad yes

RoxygenNote 6.0.1.9000

NeedsCompilation no

Repository CRAN

Date/Publication 2018-07-16 14:00:02 UTC

R topics documented:

Arothron-package	2
aro.clo.points	3
bary.mesh	3
compare_check.set	4
dec.curve	5
DM_base_sur	6

DM_face_sur	6
DM_set	7
dta	7
endo_set	9
export_amira	9
export_amira.path	10
ext.int.mesh	10
ext.mesh.rai	11
grid_pov	12
human_skull	12
krd1_tooth	13
landmark_frm2amira	13
malleus_bone	14
MAs_sets	14
noise.mesh	14
out.inn.mesh	15
patches_frm2amira	16
pov_selector	17
read.amira.dir	18
read.amira.set	18
read.path.amira	19
repmat	19
RMs_sets	20
SCPI.mesh	20
sinus_set	20
SM_set	21
spherical.flipping	21
trasf.mesh	22
yoda_set	22
yoda_sur	23
Index	24

Arothron-package

Geometric Morphometrics Analyses

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, and functions to import and export the coordinates of landmarks and 3D paths into the 'landmarkAscii' and 'am' format files.

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

aro.clo.points	<i>aro.clo.points</i>
----------------	-----------------------

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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)
```

bary.mesh	<i>bary.mesh</i>
-----------	------------------

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Examples

```
## 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)
rgl.close()
## Create and plot the new 3D curve (with intermediate points)
dec_curve_3D<-dec.curve(curve_3D, 2, plot = TRUE)
```

DM_base_sur

example dataset

Description

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

Usage

```
data(DM_base_sur)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

DM_face_sur

example dataset

Description

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

Usage

```
data(DM_face_sur)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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)
```

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

Description

POVs defined inside the endocranial cavity

Usage

```
data(endo_set)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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 *export_amira.path*

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Examples

```
library(Arothron)
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 *ext.int.mesh*

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 FALSE 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
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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

ext.mesh.raï

ext.mesh.raï

Description

This function returns a 3D mesh with colours based on the vertices visibile 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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

grid_pov

grid_pov

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

human_skull

example dataset

Description

3D mesh of a human skull

Usage

```
data(human_skull)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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

Description

3D mesh of a deciduous Neanderthal tooth

Usage

```
data(krd1_tooth)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

malleus_bone *example dataset*

Description

3D mesh of a human malleus

Usage

```
data(malleus_bone)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

MAs_sets *example dataset*

Description

Landmark configurations of the manual alignments

Usage

```
data(MAs_sets)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

noise.mesh *noise.mesh*

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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=1.0, 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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

pov_selector	<i>pov_selector</i>
--------------	---------------------

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

read.amira.dir	<i>read.amira.dir</i>
----------------	-----------------------

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

read.amira.set	<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

name.file	character: path of a landmarkAscii file
nland	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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

`read.path.amira` *read.path.amira*

Description

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

Usage

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

Arguments

`path.name` character: path of surface path .ascii extension file

Value

data numeric: a kxd matrix with xyz coordinates

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

`repmat` *repmat*

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

RMs_sets

example dataset

Description

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

Usage

```
data(RMs_sets)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

SCP1.mesh

example dataset

Description

Mesh of the Saccopastore 1 Neanderthal skull

Usage

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

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

sinus_set

example dataset

Description

POVs sampled inside the maxillary sinus cavity

Usage

```
data(sinus_set)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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

Description

Landmark configuration associated to the starting model

Usage

```
data(SM_set)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

spherical.flipping	<i>spherical.flipping</i>
--------------------	---------------------------

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

trasf.mesh	<i>trasf.mesh</i>
------------	-------------------

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, Alessio Veneziano, Marina Melchionna, Pasquale Raia

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

Description

Landmark set on Yoda

Usage

```
data(yoda_set)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

yoda_sur

example dataset

Description

Mesh of Yoda

Usage

`data(yoda_sur)`

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Index

*Topic **Arothron**

- DM_base_sur, [6](#)
- DM_face_sur, [6](#)
- DM_set, [7](#)
- endo_set, [9](#)
- human_skull, [12](#)
- krd1_tooth, [13](#)
- malleus_bone, [14](#)
- MAs_sets, [14](#)
- RMs_sets, [20](#)
- SCP1.mesh, [20](#)
- sinus_set, [20](#)
- SM_set, [21](#)
- yoda_set, [22](#)
- yoda_sur, [23](#)

aro.clo.points, [3](#)

Arothron (Arothron-package), [2](#)

Arothron-package, [2](#)

bary.mesh, [3](#)

compare_check.set, [4](#)

dec.curve, [5](#)

DM_base_sur, [6](#)

DM_face_sur, [6](#)

DM_set, [7](#)

dta, [7](#)

endo_set, [9](#)

export_amira, [9](#)

export_amira.path, [10](#)

ext.int.mesh, [10](#)

ext.mesh.rai, [11](#)

grid_pov, [12](#)

human_skull, [12](#)

krd1_tooth, [13](#)

landmark_frm2amira, [13](#)

malleus_bone, [14](#)

MAs_sets, [14](#)

noise.mesh, [14](#)

out.inn.mesh, [15](#)

patches_frm2amira, [16](#)

pov_selector, [17](#)

read.amira.dir, [18](#)

read.amira.set, [18](#)

read.path.amira, [19](#)

repmat, [19](#)

RMs_sets, [20](#)

SCP1.mesh, [20](#)

sinus_set, [20](#)

SM_set, [21](#)

spherical.flipping, [21](#)

trasf.mesh, [22](#)

yoda_set, [22](#)

yoda_sur, [23](#)