Package 'molaR'

May 9, 2026

Title Dental Surface Complexity Measurement Tools

Version 6.0

Description Surface topography calculations of Dirichlet's normal energy, relief index, surface slope, and orientation patch count for teeth using scans of enamel caps.
Importantly, for the relief index and orientation patch count calculations to work, the scanned tooth files must be oriented with the occlusal plane parallel to the x and y axes, and perpendicular to the z axis. The files should also be simplified, and smoothed in some other software prior to uploading into R.

Depends R (>= 2.10), alphahull, Rvcg, pracma, htmltools

License GPL

LazyData true

Encoding UTF-8

Suggests knitr, rmarkdown, rgl

VignetteBuilder knitr

NeedsCompilation no

Author James D. Pampush [aut, cre, cph],
Paul E. Morse [aut, cph],
Alexander Q. Vining [aut, cph],
Edward Fuselier [aut, cph]

Maintainer James D. Pampush <jdpampush@gmail.com>

RoxygenNote 7.3.3

Repository CRAN

Date/Publication 2025-11-08 07:10:02 UTC

Contents

ARC *Calculate several measures of Area Relative Curvature*

Description

A function that calculates the average slope over a tooth or some other 3D surface

Usage

```
ARC(plyFile, BoundaryDiscard = "Vertex", Range = c(0.01, 0.99))
```

Arguments

`plyFile`	An object of classes 'mesh3d' and 'shape3d' with calculated normals
`BoundaryDiscard`	String indicating how to handle the exclusion of boundary faces. Default of Vertex excludes faces which have at least 1 vertex on the boundary
`Range`	A pair of values which set lower and upper outlier exclusions.

Details

The function requires an object created by reading in a ply file.

This function calculates Area Relative Curvature, as described by Guy et al. (2013)

Examples

```
arc_output <- ARC(Tooth)
summary(arc_output)
```

ARC3d *Plot Area Relative Curvature (ARC) on a surface (HTML widget; continuous bar legend)*

Description

Creates a three-dimensional rendering of Area Relative Curvature (ARC) on a mesh and returns an HTML widget with a continuous, vertical color-bar legend. Curvature is split into four negative and four positive bands (quantiles on each side of zero) for per-face coloring. The legend uses a single blended bar assembled from the same eight colors used on the surface, with negative values at the bottom and positive values at the top.

Usage

```
ARC3d(
  ARC_object,
  main = "",
  cex = 1,
  colors = c("darkblue", "blue", "powderblue", "gray", "gray", "tan", "orange",
    "darkorange1"),
  fieldofview = 0,
  legend = TRUE,
  widget_size_px = 768,
  scene_zoom = 1.5,
  leftOffset = 0,
  title_font_size_px = 30,
  legend_magnify = 1,
  legendTextCol = "black",
  legendLineCol = "black",
  fileName = NA,
  binary = FALSE
)
```

Arguments

ARC_object	An object that stores the output of ARC(), including $Mean_Face_Curvature and $plyFile (an rgl::mesh3d).
main	Character; plot title (default "")
cex	Numeric; relative size multiplier for legend text (default 1)
colors	Character vector of length >= 8 with the eight display colors for the curvature bands. Defaults match the classic ARC3d: c("darkblue","blue","powderblue","gray","gray","tan","orange These eight colors are used both for the surface and for the legend bar.

fieldofview	Numeric; field of view in degrees; 0 = isometric (default 0)
legend	Logical; show legend panel (default TRUE)
widget_size_px	Integer; square size of the 3D widget in pixels (default 768)
scene_zoom	Numeric; initial zoom on the 3D scene (default 1.5)
leftOffset	Numeric (-1..1); small horizontal camera nudge (default 0)
title_font_size_px	Integer; title font size in pixels (default 30)
legend_magnify	Numeric; additional legend scale factor (default 1)
legendTextCol	Legend text color (default "black")
legendLineCol	Legend border color for the bar box (default "black")
fileName	Character or NA; if non-NA, write a colorized *.ply to disk (default NA)
binary	Logical; write PLY as binary (default FALSE, i.e., ascii)

Details

Run ARC() first to compute per-face ARC values.

Binning scheme: The mean face curvature values are split by sign. For negative values, the empirical quartiles (Q1..Q3) define three internal boundaries, creating four bins below zero; the same is done for positive values above zero. The eight bins map to the eight provided colors. The legend bar is a continuous CSS gradient built from those same colors so that the bar visually matches the surface and blends smoothly from one band to the next. Tick labels are shown at LowQs, 0, and HighQs (top to bottom = high to low).

Value

An htmltools::browsable object containing the title, 3D widget, and legend.

Examples

```
# ARC_out <- ARC(Tooth)
if(interactive()){ARC3d(ARC_out, main = "Area Relative Curvature")}
```

Check2D *Visual check of 2D footprint triangles (HTML widget; no Quartz)*

Description

Plot the 2D footprint triangles and their vertices used in the RFI 2D area calculation, as a visual QA step to detect spurious triangles inside the footprint. The result is an HTML widget (no Quartz/X11 window) with an optional legend, suitable for RStudio Viewer, Quarto/HTML, and browsers.

Usage

```
Check2D(
  RFI_Output,
  FootColor = "red",
  TriPointsColor = "black",
  main = "",
  cex = 1,
  legend = TRUE,
  widget_size_px = 768,
  scene_zoom = 1.5,
  leftOffset = 0,
  fieldofview = 0,
  title_font_size_px = 30,
  legend_magnify = 1,
  point_size = 5,
  alpha = 0.35
)
```

Arguments

`RFI_Output`	An object that stores the output of `RFI()`
`FootColor`	Character; fill color for the footprint triangles (default `"red"`)
`TriPointsColor`	Character; point color for footprint triangle vertices (default `"black"`)
`main`	Character; plot title (default `""`)
`cex`	Numeric; relative size multiplier for legend text (default `1`)
`legend`	Logical; show a right-hand legend (default `TRUE`)
`widget_size_px`	Integer; square size of the widget in pixels (default `768`)
`scene_zoom`	Numeric; initial zoom of the 3D scene (default `1.5`)
`leftOffset`	Numeric (-1..1); small horizontal camera nudge (default `0`)
`fieldofview`	Numeric; field of view in degrees; 0 = isometric (default `0`)
`title_font_size_px`	Integer; title font size in pixels (default `30`)
`legend_magnify`	Numeric; additional legend scale factor (default `1`)
`point_size`	Numeric; size for triangle vertex points (default `5`)
`alpha`	Numeric between 0 and 1; transparency for the footprint fill (default `0.35`)

Details

This reproduces the same intent as the classic `Check2D()`: it draws the triangles from `RFI_Output$Footprint_Triangles` and the corresponding 2D coordinates in `RFI_Output$Flattened_Pts`. The triangles are plotted in the **XY plane** (z = 0). If you see points/triangles **inside** the footprint that shouldn't be there, it typically indicates an **alpha** value that is too small in the RFI step, leading to an inflated 2D footprint.

The function uses a headless rgl device and returns an `rglwidget` embedded in an HTML layout (title atop, legend on the right)—matching the approach used in `DNE3d()`.

Value

An htmltools-browsable object containing the title, 3D widget, and legend.

Examples

```
# RFI_out <- RFI(Tooth, alpha = 0.5)
# Check2D(RFI_out, FootColor = "tomato", TriPointsColor = "black",
# main = "RFI 2D Footprint QA")
```

DNE *Calculate Dirichlet normal energy of a surface*

Description

A function that calculates Dirichlet normal energy following the method of Bunn et al. (2011) Comparing Dirichlet normal surface energy of tooth crowns, a new technique of molar shape quantification for dietary inference, with previous methods in isolation and in combination. Am J Phys Anthropol 145:247-261 doi: 10.1002 ajpa.21489

Usage

```
DNE(plyFile, outliers = 0.1, kappa = 0, BoundaryDiscard = "Vertex", oex = "c")
```

Arguments

`plyFile`	An object of classes 'mesh3d' and 'shape3d' with calculated normals
`outliers`	The percentile of Dirichlet energy density values to be excluded defaults to top 0.1 percent
`kappa`	An integer value of mean curvature to define concave vs convex faces
`BoundaryDiscard`	String indicating how to handle the exclusion of boundary faces. Default of Vertex excludes faces which have at least 1 vertex on the boundary
`oex`	String indicating outlier exclusion principle. Defaults to 'c', which combines all convex and concave faces and removes the percentage of outliers defined by `outliers`. See details.

Details

The function requires an object created by reading in a ply file.

Dirichlet normal energy is calculated on meshes that represent specimen surfaces and have already been simplified and pre-smoothed in a 3D data editing program.

In the default settings, the function seeks to discard boundary faces. This can be changed by adjusting the `BoundaryDiscard` argument to 'None' which will not discard any faces on the boundary. Further, there are two ways of excluding boundary faces. Either if they have a leg on the boundary by setting `BoundaryDiscard='Leg'` or by excluding any face which has a vertex on the boundary

with BoundaryDiscard='Vertex'. The function defaults to remove the top 0.1 percent of calculated energy densities as outliers. Mesh orientation does not affect this calculation.

Faces are labeled as concave or convex on the basis of the kappa value, which defaults to 0. Each face is assigned a kappa value, which describes the the localized degree of convergence or divergence among the three vertex normals on each face. Faces with positive kappa values have vertex normals that are divergent. Faces with negative kappa values possess vertex normals that are convergent. Users can adjust the kappa value to redefine areas of the tooth assigned to the convex or concave bin.

The mode of Outlier exclusion can be modified with the oex argument. The default, oex='c' considers the Dirichlet energy density values of all faces on the surface and removes those in the top percentile defined by outliers, regardless of the convexity or concavity bins. The alternative, oex='s', divides the surface into concave and convex portions, then removes the percentile defined by outliers from each of these subsets before calculating total surface DNE.

Examples

```
DNE_output <- DNE(Tooth)
summary(DNE_output)
```

DNE3d *Plot results of a DNE analysis of a surface*

Description

A molaR surface plotting function.

Usage

```
DNE3d(
  DNE_File,
  setMax = 0,
  logColors = TRUE,
  signColor = TRUE,
  concaveMute = FALSE,
  cuttingFocus = FALSE,
  main = "",
  cex = 1,
  legend = TRUE,
  widget_size_px = 768,
  scene_zoom = 1.5,
  leftOffset = 0,
  fieldofview = 0,
  title_font_size_px = 30,
  legend_magnify = 1,
  fileName = NA,
  binary = FALSE
)
```

Arguments

DNE_File	An object that stores the output of the DNE() function
setMax	User-defined upper range for plotting color scheme, see Details
logColors	Logical that log transforms the color scheme
signColor	Logical indicating whether or not to plot by concavity vs convexity. Plotting by curve orientation is the default.
concaveMute	Logical indicating whether or not to mute the concave portion of the Dirichlet normal density coloration. Default is FALSE.
cuttingFocus	Logical indicating whether or not to mute the concave portion and faces with Dirichlet normal density below the top quartile. Default is FALSE.
main	String indicating plot title
cex	Numeric setting the relative size of the legend
legend	Logical indicating whether or not a legend should be displayed
widget_size_px	Sets the plot size in pixels. Default is 768 and always square
scene_zoom	Set the initial viewing window of the 3d scene. Default is 1.5
leftOffset	Numeric between -1 and 1 setting the amount of offset for the plotted surface to the left. Larger values push surface farther to right.
fieldofview	Passes an argument to par3d() changing the field of view (in degrees) of the resulting 3D plot
title_font_size_px	Sets the title font size in pixels. Default is 30
legend_magnify	Default set to 1, changes the scaling on legend.
fileName	String indicating a name to save the plotted surface to as a *.ply file; default of 'NA' will not save a file
binary	Logical indicating whether or not the saved surface plot should be binary, passed to vcgPlyWrite()

Examples

```
## Not run:
DNE_output <- DNE(Tooth)
DNE3d(DNE_output)

## End(Not run)
```

`DNE3dDiscard` *Plot discarded-face results of a DNE analysis (HTML widget; no Quartz)*

Description

Produces a three-dimensional rendering that highlights the faces discarded from the DNE calculation—boundary faces and statistical outliers—along with optional concavity marking. The plot is returned as an HTML widget with a simple, readable legend panel to the right.

Usage

```
DNE3dDiscard(
  DNE_File,
  baseCol = "gray",
  boundCol = "red",
  outlierCol = "lawngreen",
  concaveCol = baseCol,
  main = "",
  cex = 1,
  legend = TRUE,
  widget_size_px = 768,
  scene_zoom = 1.5,
  leftOffset = 0,
  fieldofview = 0,
  title_font_size_px = 30,
  legend_magnify = 1,
  legendTextCol = "black",
  legendLineCol = "black",
  fileName = NA,
  binary = FALSE
)
```

Arguments

`DNE_File`	An object that stores the output of `DNE()`
`baseCol`	Base color for typical faces (default `"gray"`)
`boundCol`	Color for boundary faces discarded from DNE (default `"red"`)
`outlierCol`	Color for faces discarded as outliers (default `"lawngreen"`)
`concaveCol`	Color for concave faces; when left equal to `baseCol` the concave category is not distinguished or shown in the legend (default = `baseCol`)
`main`	Character; plot title (default `""`)
`cex`	Numeric; relative size multiplier for legend text (default `1`)
`legend`	Logical; show legend panel (default `TRUE`)
`widget_size_px`	Integer; square size of the 3D widget in pixels (default `768`)

scene_zoom	Numeric; initial zoom on the 3D scene (default 1.5)
leftOffset	Numeric (-1..1); small horizontal camera nudge (default 0)
fieldofview	Numeric; field of view in degrees; 0 = isometric (default 0)
title_font_size_px	Integer; title font size in pixels (default 30)
legend_magnify	Numeric; additional legend scale factor (default 1)
legendTextCol	Color for legend text (default "black")
legendLineCol	Color for legend swatch borders/lines (default "black")
fileName	Character or NA; if non-NA, write a colorized *.ply to disk (default NA)
binary	Logical; write PLY as binary (default FALSE, i.e., ascii)

Details

Run DNE() first to compute per-face values and discard sets.

What this shows

- *Boundary faces* (e.g., with >=1 boundary vertex) and *outlier faces* (top extreme tail) that were excluded from DNE are colored distinctly; all other faces get baseCol. If concaveCol differs from baseCol, concave regions are tinted accordingly and labeled in the legend.

HTML widget

- Uses a headless rgl device and returns an rglwidget embedded beside a simple HTML legend (no Quartz window). Layout and centering mirror the updated DNE3d.

Saving a PLY

- If fileName is provided, a colorized PLY is written. For ascii PLY, a comment line noting the generator/version is inserted (as in the updated DNE3d).

Value

An htmltools-browsable object containing the title, 3D widget, and legend.

Examples

```
# DNE_out <- DNE(Tooth)
if(interactive()){DNE3dDiscard(DNE_out, main = "DNE -- Discarded Faces")}
```

DNEbar *Plot advanced results of a DNE surface analysis*

Description

a molaR plotting function

Usage

```
DNEbar(
  DNE_File,
  main = "",
  convexCol = "hotpink",
  concaveCol = "deepskyblue",
  type = "both",
  legendPos = "topright",
  legendInset = 0,
  las = 1,
  names.arg = "",
  cex.names = 1
)
```

Arguments

DNE_File	An object that stores the output of the DNE() function
main	User's title for plot.
convexCol	Color for the convex DNE total. Default='hotpink'
concaveCol	Color for the concave DNE total. Default='deepskyblue'
type	string to determine what parameters to plot. Default=both and both concave and convex DNE totals will be plotted in stacked bar plot. See details
legendPos	string to determine location of the legend. Default='topright' see details.
legendInset	numeric value determining how far to inset the legend from plot boarder. Default=0
las	logical indicating orientation of the x-axis labels for each bar plot. Enter either 1 or 2.
names.arg	concatenated string of surface names for labels. If none supplied function will pull names from the object itself.
cex.names	Font size for the bar labels. Default is 1.

Details

This function creates a stacked barplot of DNE values. It colors them according to curve orientation, which is defined by the kappa parameter in DNE() function. If multiple DNE objects are grouped together the barplot will return a set. When employed on a single DNE object this will return a single stacked bar.

The argument type accepts either 'Concave' or 'Convex' to plot only concave or convex DNE totals respectively. Default=NA and results in both totals being plotted in stacked barplot.

The argument legendPos is a string that determines the position of the legend. Default='topright' but will accept any of the following keywords: 'bottomright', 'bottom', 'bottomleft', 'left', 'topleft', 'top', 'topright', 'right', or 'center'.

Examples

```
DNEs <- list()
DNEs$Tooth <- DNE(Tooth)
DNEs$Hills <- DNE(Hills)
DNEbar(DNEs)
```

DNEDensities *Plot advanced results of a DNE surface analysis*

Description

Plot advanced results of a DNE surface analysis

Usage

```
DNEDensities(
  DNE_File,
  main = "",
  type = "DNE",
  legendPos = "topright",
  convexCol = "hotpink",
  concaveCol = "deepskyblue"
)
```

Arguments

DNE_File	An object that stores the output of the DNE function
main	User's title for plot. Default is blank
type	string determining which density plots to make. Default
legendPos	string to determine location of the legend. Default='topright' see details. is to plot DNE face densities. Alternatively can plot face areas with 'area'
convexCol	Color for the convex density polygon, Default='hotpink'
concaveCol	Color for the concave density polygon, Default='deepskyblue'

Details

This function creates a set of overlapping density plots of two potential types. The user can plot overlapping density plots that sort the surface into concave and convex portions for plotting. The function will default to plotting DNE density values, however density of face surface areas sorted into concave and convex portions of the surface can be plotted by calling type='area'. Colors can be customized by altering the convexCol and concaveCol arguments.

Examples

```
DNE_output <- DNE(Tooth)
DNEDensities(DNE_output)
```

DNEpie *Plot advanced results of a DNE surface analysis*

Description

Plot advanced results of a DNE surface analysis

Usage

```
DNEpie(
  DNE_File,
  main = "",
  type = "area",
  convexCol = "hotpink",
  concaveCol = "deepskyblue"
)
```

Arguments

DNE_File	An object that stores the output of the DNE() function
main	User's title for the plot
type	string determine which parameters to plot. Default='DNE' also accepts 'area' to plot pie charts of the area.
convexCol	Color for the portion of the pie chart representing convex contribution. Default='hotpink'. Accepts any color keyword.
concaveCol	Color for the portion of the pie chart representing concave contribution. Default='deepskyblue'. Accepts any color keyword.

Details

This function creates a pie chart of the total area or DNE of the surface originating from the concave or convex portions of the surface. The function defaults to plotting surface area, however, relative proportion of total DNE from the concave and convex portions of the surface can be plotted by calling type='DNE'. Colors can be customized by altering the convexCol and concaveCol arguments.

Examples

```
DNE_output <- DNE(Tooth)
DNEpie(DNE_output)
```

Hills *Hills surface mesh*

Description

Sample mesh created with the formula: 'z=3cos(x/2)+3sin(y/2)

A triangular mesh representing a sine-cosine plane - called by data(Hills)

Usage

```
data(Hills)
```

Format

An object of class "mesh3d"

Hills: triangular mesh representing a sine-cosine plane.

Examples

```
data(Hills)
DNE_Output <- DNE(Hills)
DNE3d(DNE_Output)
```

molaR_Batch *Run molaR analyses on a batch of specimens*

Description

A function that automates molaR analyses on multiple specimens. Several different analyses can be performed on each surface, with specifications for analysis parameters.

Usage

```
molaR_Batch(
  pathName = getwd(),
  fileName = "molaR_Batch.csv",
  DNE = TRUE,
  RFI = TRUE,
  OPCr = TRUE,
  OPC = FALSE,
  Slope = TRUE,
  details = TRUE,
  parameters = TRUE,
  ...
)
```

Arguments

pathName	The path to the folder containing all ply surfaces to be analyzed. Defaults to the working directory.
fileName	Name for the output .csv file containing results and parameters
DNE	Logical indicating whether or not to perform the DNE calculation
RFI	Logical indicating whether or not to perform the RFI calculation
OPCr	Logical indicating whether or not to perform the OPCr calculation
OPC	Logical indicating whether or not to perform the OPC calculation
Slope	Logical indicating whether or not to perform the Slope calculation
details	Logical indicating whether or not to save additional output from some of the topographic analyses
parameters	Logical indicating whether or not to save the list of analysis parameters used in the batch run
...	Additional arguments passed to the topographic analysis functions. See Details.

Details

This function allows a user to set the analyses from molaR they want to run on a batch of ply files. Output is saved to a csv file. By default, the batch function will perform specified analyses on all ply files in the working directory. A different folder can be specified with pathName. Output saves as .csv to the folder that contains the analyzed ply files.

Any of the default arguments of the various topographic analysis functions can be modified for the batch by specifying them when calling molaR_Batch, e.g., the DNE kappa value can be changed to 'X' by specifying kappa = X. Users are **strongly** encouraged to review the documentation for DNE(), RFI(), OPCr(), OPC(), and Slope() and to understand the effects of alterations before making changes. A recommended practice for analyzing RFI in a batch of specimens is to enable findAlpha = TRUE given that the ideal alpha value is likely to vary among different specimens. However, this will increase calculation time (see documentation for RFI).

By default, the batch output will retain some additional details of the analysis. These include, in the case of DNE: convex and concave DNE values, convex and concave surface areas; in the case of RFI: 3D and 2D surface areas and analysis alpha values; in the case of OPCr: the surface OPC value calculated at each rotation; and in the case of OPC: the patch count for each bin. These results will be discarded and only the final result of each topographic analysis will be retained if details = FALSE.

The function will save a list of all parameters used in all batch analyses to the output .csv file, below the results. This can be suppressed with parameters = FALSE, but is recommended as a check on how analyses were performed when returning to results in the future. If the function is assigned to an object in R, the parameters are not included in the resultant data.frame, but will still be included in the .csv file by default.

Note that batch processing updates will not display by default if using RGui for Windows. Disable Misc -> Buffered output (Ctrl+W) if you wish to view batch processing progress in RGui for Windows.

molaR_Clean *Clean up problem ply files*

Description

Function will remove floating verticies, and faces with zero area. These can cause issues when using molaR's primary functions of DNE, RFI, and OPC

Usage

```
molaR_Clean(plyFile, cleanType = "Both", verbose = TRUE)
```

Arguments

plyFile	An object of classes 'mesh3d' and 'shape3d'
cleanType	String with three arguments defining what to clean: Vertices, Faces, or Both. Defaults to Both.
verbose	Logical indicating if the function should report changes to ply

Details

This function cleans up problematic ply files. Some smoothed files will have faces of zero area, or floating vertices. DNE and OPC cannot be calculated on these files. Running the plys through this function will allow those calculations to be made.

Examples

```
Tooth <- molaR_Clean(Tooth)
```

OPC *Calculate orientation patch count of a surface*

Description

A function that bins patches of a mesh surface that share general orientation and sums the number of unique patches given certain parameters Modified into 3D from the original 2.5D method described by Evans et al. (2007) High-level similarity of dentitions in carnivorans and rodents. Nature 445:78-81 doi: 10.1038/nature05433

Usage

```
OPC(plyFile, rotation = 0, minimum_faces = 3, minimum_area = 0)
```

Arguments

plyFile	An object of classes "mesh3d" and "shape3d" with calculated vertex normals
rotation	Rotates the file in degrees about the center vertical axis
minimum_faces	Minimum number of ply faces required for a patch to be counted towards the total patch count
minimum_area	Minimum proportion (100%=1.0) of total surface area a patch must occupy to be counted towards the total patch count

Details

The function requires a mesh object created by reading in a ply file utilizing either the, vcgPlyRead function.

Orientation patch count is calculated on meshes that represent specimen surfaces and have already been downsampled to 10,000 faces and pre-smoothed in a 3D data editing program. Alignment of the surface will have a large effect on patch orientation and must be performed in a 3D data editing program such as Avizo. The occlusal surface of the specimen must be made parallel to the X- and Y-axes and perpendicular to the Z-axis.

The default for minimum_faces is to ignore patches consisting of two or fewer faces on the mesh. Changing the minimum_area value will disable minimum_faces.

Examples

```
OPC_output <- OPC(Tooth)
summary(OPC_output)
```

OPC3d — *Plot results of OPC analysis of a surface (3D HTML widget; numbered legend; distinct colors)*

Description

Produces a three-dimensional rendering of face orientation on a surface and returns an HTML widget with a clean, numbered pie legend Wedges can be scaled by patch counts. Default colors emphasize clarity.

Usage

```
OPC3d(
  OPC_File,
  binColors = NULL,
  patchOutline = FALSE,
  outlineColor = "black",
  maskDiscard = FALSE,
  legend = TRUE,
  main = "",
```

```
  cex = 1,
  scaleLegend = FALSE,
  legendTextCol = "black",
  legendLineCol = "black",
  leftOffset = 0,
  fieldofview = 0,
  widget_size_px = 768,
  scene_zoom = 1.5,
  title_font_size_px = 30,
  title_font_family =
    "system-ui, -apple-system, Segoe UI, Roboto, Helvetica, Arial, sans-serif",
  legend_magnify = 1,
  fileName = NA,
  binary = FALSE
)
```

Arguments

OPC_File	An object that stores the output of OPC()
binColors	Optional character vector of colors for each directional bin. If not supplied, a high-contrast set is generated automatically (Okabe–Ito <= 12 bins; vivid HCL rainbow for more).
patchOutline	Logical; draw patch boundary outlines on the mesh (default FALSE)
outlineColor	Color for patch outlines (default "black")
maskDiscard	Logical; color discarded/small patches black in the mesh (default FALSE)
legend	Logical; show a right-hand legend (default TRUE)
main	Character; plot title (default "")
cex	Numeric; relative size multiplier for legend text (default 1)
scaleLegend	Logical; scale legend pie wedges by patch counts (default FALSE)
legendTextCol	Color for legend text (default "black")
legendLineCol	Color for legend ring/lines (default "black")
leftOffset	Numeric (-1..1); small horizontal camera nudge (default 0)
fieldofview	Numeric; field of view in degrees; 0 = isometric (default 0)
widget_size_px	Integer; square size of the 3D widget in pixels (default 768)
scene_zoom	Numeric; initial zoom on the 3D scene (default 1.5)
title_font_size_px	Integer; title font size in pixels (default 30)
title_font_family	CSS font-family for title/legend (default system UI stack)
legend_magnify	Numeric; additional legend scale factor (default 1)
fileName	Character or NA; if non-NA, write a colored *.ply to disk (default NA)
binary	Logical; write PLY as binary (default FALSE, i.e., ascii)

Details

Run `OPC()` first to compute patches and directional bins.

Legend Wedges are **numbered (1..N)** and can be **scaled** to patch counts.

Saving a PLY When `fileName` is provided, a colorized PLY is written. Colors are assigned per face via a vertex-duplication strategy so external tools (e.g., MeshLab) show the same appearance. For ascii PLY, a comment line notes the generator/version.

Value

- **Interactive**: an `htmltools`-browsable object (title, 3D widget, legend).
- **Non-interactive**: (invisibly) a list with class `"OPC3d_spec"` containing: `title`, `bin_colors`, `face_colors`, `bin_sizes`, `bin_labels`, `legend`, `scaled`.

Examples

```
# OPC_out <- OPC(Tooth)
if (interactive()) {
  OPC3d(OPC_out, legend = TRUE, scaleLegend = TRUE,
        main = "OPC -- Numbered Bins")
}
```

OPCbinareas *Visualize surface area distribution into separate OPC orientation bins.*

Description

This function will make either a bar plot or pie chart showing the surface area assigned to each OPC orientation bin.

Usage

```
OPCbinareas(
  OPC_File,
  main = "",
  binColors = hsv(h = (seq(10, 290, 40)/360), s = 0.9, v = 0.85),
  type = "bar"
)
```

Arguments

`OPC_File`	An object that stores the output of an OPC analysis using `OPC()`.
`main`	Title for plot.
`binColors`	Allows the user to define the fill colors for each directional bin. see details
`type`	String argument to determine type of plot, either bar' or 'pie'. Default is set to 'bar'

Details

This function will create either bar or pie chats visualising the distribution of surface area into each of the OPC orientation bins. Colors can be customized but are meant to match the default settings in the `OPC3d()` function.

Examples

```
OPC_Object <- OPC(Tooth)
OPCbinareas(OPC_Object)
```

`OPCr` *Calculate average orientation patch count after several rotations*

Description

A function that calls OPC iteratively after rotating mesh a selected number of degrees around the Z-axis following Evans and Jernvall (2009) Patterns and constraints in carnivoran and rodent dental complexity and tooth size. J Vert Paleo 29:24A

Usage

```
OPCr(plyFile, steps = 8, stepSize = 5.625, minimum_faces = 3, minimum_area = 0)
```

Arguments

`plyFile`	An object of classes 'mesh3d' and 'shape3d' with calculated normals
`steps`	Number of iterations to run the `OPC()` function on the mesh
`stepSize`	Amount of rotation (in degrees) about the Z-axis to adjust mesh surface by between each iteration of `OPC()`
`minimum_faces`	Argument to pass to the `OPC()` function
`minimum_area`	Argument to pass to the `OPC()` function

Details

The function requires a mesh object created by reading in a ply file utilizing either the, `vcgPlyRead` function.

Default number of steps is 8, with a stepSize of 5.625 degrees, following the original published definition of OPCR.

See the details for the `OPC` function for more information about preparing mesh surfaces and the effects of `minimum_faces` and `minimum_area`.

OPCr_Example1 *OPCr_Example1 - object created by OPCr function used as an example.*

Description

This object is needed to pass the CRAN upload requirements and still keep the vignette.

Format

OPCr_Example1: molaR produced object.

OPCr_Example2 *OPCr_Example2 - object created by OPCr function used as an example.*

Description

This object is needed to pass the CRAN upload requirements and still keep the vignette.

Format

OPCr_Example2: molaR produced object.

plyPlaneCut *Cut a PLY Mesh Along a Specified Plane (HTML widget; no Quartz)*

Description

Cuts a mesh by a plane and returns either one side or both sides of the cut. The result is also previewed as an HTML widget (no Quartz/X11) with the kept side in solid color, the other side semi-transparent, and the cutting plane shown.

Usage

```
plyPlaneCut(
  plyFile,
  axis = "Z",
  vertIndex = NA,
  cut_value = NA,
  keepBoth = FALSE,
  plane = NA,
  flipAxis = FALSE,
  display = TRUE,
```

```
  kept_col = "#3C8DFF",
  other_col = "gray70",
  plane_col = "firebrick",
  plane_alpha = 0.35,
  main = "",
  widget_size_px = 768,
  scene_zoom = 1.5,
  leftOffset = 0,
  fieldofview = 0,
  title_font_size_px = 30,
  title_font_family =
    "system-ui, -apple-system, Segoe UI, Roboto, Helvetica, Arial, sans-serif"
)
```

Arguments

plyFile	An object of class 'mesh3d'.
axis	Character, one of "X","Y","Z"; used when deriving the plane from a vertex index or a numeric cut value (ignored when 'plane' is supplied). Default "Z".
vertIndex	Integer index of a mesh vertex used to place an orthogonal plane through that vertex along 'axis'. Ignored if 'plane' or 'cut_value' is supplied.
cut_value	Numeric coordinate along 'axis' at which to place the orthogonal plane. Ignored if 'plane' is supplied. (HTML-friendly alternative to interactive pick)
keepBoth	Logical; if TRUE and the plane intersects the mesh, return both sides as a list with elements 'meshA' and 'meshB'. Default FALSE.
plane	Numeric vector c(a,b,c,d) giving a plane in ax + by + cz + d = 0 form. If provided, overrides 'axis', 'vertIndex', and 'cut_value'.
flipAxis	Logical; reverse the plane normal direction used to decide which side is kept when keepBoth = FALSE. Default FALSE.
display	Logical; if TRUE, render an HTML widget preview. Default TRUE.
kept_col	Color for the kept portion in the preview (default "#3C8DFF").
other_col	Color for the other portion in the preview (default "gray70").
plane_col	Color for the plane polygon in the preview (default "firebrick").
plane_alpha	Alpha for the plane polygon (default 0.35).
main	Plot title (default "")
widget_size_px	Square widget size in pixels (default 768)
scene_zoom	Initial zoom for the 3D scene (default 1.5)
leftOffset	Horizontal camera nudge (-1..1 recommended) (default 0)
fieldofview	Field of view in degrees (0 = isometric) (default 0)
title_font_size_px	Title font size in pixels (default 30)
title_font_family	CSS font-family list for the title (default system UI stack)

Details

This HTML-widget version avoids opening a native rgl window and mirrors the device-and-layout pattern you used in your updated `DNE3d`. Interactive vertex picking (`select3d`) isn't supported in widget mode; supply the cut as `plane`, `vertIndex` + `axis`, or `cut_value` + `axis`. The kept side selection is controlled by the normal direction and `flipAxis`.

Value

If keepBoth = FALSE, a 'mesh3d' for the retained side. If keepBoth = TRUE, a list with elements 'meshA' and 'meshB'. The returned object also carries an attribute 'widget' with the `htmltools::browsable` preview when `display = TRUE`.

RFI *Calculate relief index for a surface*

Description

A function that calculates relief index following Boyer (2008) Relief index of second mandibular molars is a correlate of diet among prosimian primates and other mammals. J Hum Evol 55:1118-1137 doi: 10.1016/j.jhevol.2008.08.002

Usage

```
RFI(plyFile, alpha = 0.075, findAlpha = FALSE)
```

Arguments

`plyFile`	An object of classes 'mesh3d' and 'shape3d'
`alpha`	Step size for calculating the outline. See details.
`findAlpha`	Logical indicating that alpha will be auto-calculated. See details.

Details

The function requires an object created by reading in a ply file utilizing either the `vcgPlyRead` function.

Relief index is calculated by the ratio of three-dimensional surface area to two dimensional area on meshes that represent specimen surfaces and have already been pre-smoothed in a 3D data editing program. Surface alignment will have a large effect on 2D area calculation and must be performed prior to creating and reading in the ply file. The mesh must be oriented such that the occlusal plane is parallel to the X- and Y-axes and perpendicular to the Z-axis (i.e., tooth cusps pointing towards +Z).

The `alpha` parameter traces the outline of the 2D footprint. An `alpha` that is too low will result in a tracing error (returning an `"Alpha adjustment required"` message), while an `alpha` value that is too high may result in an overestimate of the 2D footprint area by failing to take into account infoldings. The user is encouraged to carefully review results using the `RFI3d` or `Check2D` functions.

Alternatively, the findAlpha argument can be used to compute an ideal alpha value for a particular PLY file for use in the RFI calculation. This is defined as the lowest value (to the nearest thousandth) returning no error or warning messages. This feature ensures accuracy, but may increase computing time significantly, depending on the number of alpha values tested. Unfortunately, there is no way to guess an appropriate alpha value a priori. After 100 unsuccessful attempts to find an appropriate alpha, the function will terminate.

The alpha value used in the calculation (whether chosen by the user or auto- computed with findAlpha) is returned in the analysis results.

Examples

```
RFI_output <- RFI(Tooth, alpha=0.5, findAlpha = FALSE)
summary(RFI_output)
```

RFI3d *Plot 3D and 2D areas of a mesh used to calculate relief index (HTML widget)*

Description

Renders a three-dimensional model of the mesh surface and a footprint of the two-dimensional area side-by-side for visual comparison. The RFI() function must be performed prior to using RFI3d().

Usage

```
RFI3d(
  RFI_File,
  displacement = -1.9,
  SurfaceColor = "gray",
  FootColor = "red",
  FootPts = FALSE,
  FootPtsColor = "black",
  Opacity = 1,
  legend = TRUE,
  main = "",
  cex = 1,
  widget_size_px = 768,
  scene_zoom = 1.5,
  leftOffset = 0,
  fieldofview = 0,
  title_font_size_px = 30,
  title_font_family =
    "system-ui, -apple-system, Segoe UI, Roboto, Helvetica, Arial, sans-serif",
  legend_magnify = 1,
  fileName = NA,
  binary = FALSE
)
```

Arguments

RFI_File	An object that stores the output of the RFI function
displacement	Numeric that moves the surface footprint some proportion of the height of the mesh. 0 is vertical center; negative values displace the footprint downward. Default -1.9.
SurfaceColor	Color for the 3D surface mesh
FootColor	Color for the 2D surface footprint
FootPts	Logical; if TRUE, plot the flattened points used for the footprint
FootPtsColor	Color of plotted footprint points when FootPts = TRUE
Opacity	Numeric from 0 to 1 controlling the opacity of the 3D surface
legend	Logical; draw a legend (right side of the widget)
main	Plot title
cex	Base scaling for legend/ticks (relative multiplier)
widget_size_px	Square widget size in pixels (default 768)
scene_zoom	Initial zoom for the 3d scene (default 1.5)
leftOffset	Horizontal camera nudge (-1..1 recommended)
fieldofview	Field of view in degrees (0 = isometric)
title_font_size_px	Title font size in pixels (default 30)
title_font_family	CSS font-family list for the title
legend_magnify	Additional scale factor for the legend text
fileName	Optional file base name to save a colored *.ply (no legend)
binary	Binary PLY if TRUE (smaller files)

Details

This function helps visualize the 3D surface area (numerator) and the 2D projected area (denominator) that comprise the relief index, by displaying both at once. Adjust Opacity to make the footprint more visible through the surface. The footprint plane can be moved along Z via displacement.

The plotting window is an HTML widget. The legend is rendered in HTML/CSS and aligned vertically at the right of the scene.

Examples

```
if(interactive()){
  rfi <- RFI(Tooth, alpha = 0.5)
  RFI3d(rfi)
}
```

`Slope` *Function to calculate the average slope of a surface*

Description

A function that calculates the average slope over a tooth or some other 3D surface

Usage

```
Slope(plyFile, Guess = FALSE)
```

Arguments

`plyFile`	An object of classes 'mesh3d' and 'shape3d' with calculated normals
`Guess`	Logical indicating whether the function should 'guess' as to the 'up' direction for the surface and to remove negative slopes from the calculation

Details

This function requires a ply file. It will calculate the slope on each face of the surface and will average the slope across the surface. This is functionally equivalent to the slope calculation used by Ungar and M'Kirera "A solution to the worn tooth conundrum in primate functional anatomy" PNAS (2003) 100(7):3874-3877

In the case of applying this function to teeth (its intended purpose), the function expects a surface with the occlusal plane normal to the Z-axis.

The `Guess` parameter is a logical asking whether or not you want the function to both guess as to the right side up of the surface, and to then discard all of the 'negative' slopes, i.e. surfaces which are over-hangs, as is frequently found on the sidewalls of teeth. If `Guess` is not engaged the mean slope will include the negative values of the overhang and will likely underestimate the average slope of the surface.

Regardless of if the `Guess` parameter is engaged, the function will also return a vector containing all of the face slope values ("Face_Slopes")

Examples

```
Slope_output <- Slope(Tooth)
summary(Slope_output)
```

`Slope3d` *Plot results of a Slope analysis of a surface (HTML widget)*

Description

A function that produces a three-dimensional HTML-based rendering of surface slope. The `Slope()` function must be performed prior to using `Slope3d()`.

Usage

```
Slope3d(
  Slope_File,
 colors = c("blue", "cornflowerblue", "green", "yellowgreen", "yellow", "orangered",
    "red"),
  maskNegatives = TRUE,
  legend = TRUE,
  main = "",
  cex = 1,
  widget_size_px = 768,
  scene_zoom = 1.5,
  leftOffset = 0,
  fieldofview = 0,
  title_font_size_px = 30,
  legend_magnify = 1,
  fileName = NA,
  binary = FALSE
)
```

Arguments

`Slope_File`	An object that stores the output of the Slope function
`colors`	Character vector of colors to build a continuous color gradient
`maskNegatives`	Logical; if TRUE, negative slopes are masked in black. If FALSE, negatives are reflected into the positive range (original behavior).
`legend`	Logical; draw a legend (right side of the widget)
`main`	Plot title
`cex`	Base scaling for legend/ticks (relative multiplier)
`widget_size_px`	Square widget size in pixels (default 768)
`scene_zoom`	Initial zoom for the 3d scene (default 1.5)
`leftOffset`	Horizontal camera nudge (-1..1 recommended)
`fieldofview`	Field of view in degrees (0 = isometric)
`title_font_size_px`	Title font size in pixels (default 30)
`legend_magnify`	Additional scale factor for the legend/tick text

fileName	Optional file base name to save a colored *.ply (no legend)
binary	Binary PLY if TRUE (smaller files)

Details

Colors represent face slope magnitudes. With maskNegatives = TRUE, negative slopes are shown in black. With maskNegatives = FALSE, negative magnitudes are reflected into the positive range for coloring continuity (original behavior).

The plotting window is an HTML widget (no Quartz/rgl window). The legend is rendered in HTML/CSS and aligned vertically at the right of the scene.

Examples

```
s <- Slope(Tooth)
if(interactive()){Slope3d(s)}
```

Tooth *Tooth a surface mesh of a tooth.*

Description

Tooth scan of USNM_112176 lower M~1~ from *Chlorocebus sp.*

A triangular mesh representing a lower M1 Chlorocebus spp. tooth - called by data(Tooth)

Usage

```
data(Tooth)
```

Format

An object of class "mesh3d"

Tooth: triangular mesh representing a sine-cosine plane.

Examples

```
data(Tooth)
DNE_Output <- DNE(Tooth)
DNE3d(DNE_Output)
```

Index