Package: ggsegExtra (via r-universe)

August 16, 2024

Title Extra Utilities for the 'ggseg' and 'ggseg3d' Plotting Tools

Version 1.5.33.004

Description Contains functions to create new data sets compatible with the 'ggseg' and 'ggseg3d' packages for plotting brain segmentations through R. Requires several external software to be installed on the system for atlas creations to work, and most of these are not available for windows. But atlas creation tools exist for 2d simple features data, and also 3d tri-surface mesh plots. Also has a list of compatible data sets and functions for easy installation of these.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

Depends R (>= 3.5.0), ggseg (>= 1.6.0), ggseg3d (>= 1.6.0)

Imports ggplot2, smoothr, raster, magick, withr, freesurfer, janitor, jsonlite, readr, dplyr, tidyr, stringr, grDevices, plotly, xml2, utils, sf, stats, rmapshaper, stars, geomorph, rgdal, pbmcapply

Suggests ggsegYeo2011, ggsegDesterieux, ggsegChen, ggsegSchaefer, ggsegGlasser, ggsegJHU, ggsegTracula, ggsegICBM, ggsegHO, ggsegDefaultExtra, ggsegDKT, knitr, rmarkdown, covr, hexSticker, testthat (>= 2.1.0), spelling

VignetteBuilder knitr

Remotes ggseg/ggsegYeo2011, ggseg/ggsegDesterieux, ggseg/ggsegChen, ggseg/ggsegSchaefer, ggseg/ggsegGlasser, ggseg/ggsegJHU, ggseg/ggsegTracula, ggseg/ggsegICBM, ggseg/ggsegHO, ggseg/ggsegDefaultExtra, ggseg/ggsegDKT

URL https://github.com/ggseg/ggsegExtra

BugReports https://github.com/ggseg/ggsegExtra/issues

Language en-US

SystemRequirements orca, FSL, Freesurfer, ImageMagick, GDAL

Repository https://ggseg.r-universe.dev

RemoteUrl https://github.com/ggseg/ggsegExtra

RemoteRef HEAD

RemoteSha c1513ba10de55c0dee4b3afed1e2436299c30d9e

Contents

aparc_2_mesh *Converts annotations to atlas*

Description

This function will create an atlas ready data.frame for ggseg3d to plot with plotly.

Usage

```
aparc_2_mesh(
  subject = "fsaverage5",
  hemisphere = "rh",
  surface = "inflated",
  annot = "aparc",
  subjects_dir = freesurfer::fs_subj_dir(),
  annot_dir = file.path(subjects_dir, subject, "label"),
  output_dir = tempdir(),
  cleanup = TRUE,
  verbose = TRUE
)
```

Arguments

subject	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
hemisphere	hemisphere, either "rh" or "lh"
surface	surface from subjects surf-folder
annot	base-name of annot file
subjects_dir	Freesurfer subject directory
annot_dir	path to directory with annot-files
output_dir	output directory path
cleanup	logical to toggle removal of all intermediary files
verbose	logical indicating to be verbose or not

Details

Based on A. Winkler scripts

Value

data frame with one row per label

Examples

```
## Not run:
dt <- aparc_2_mesh()
dt <- aparc_2_mesh(surface = "white")
dt <- aparc_2_mesh(hemisphere = "lh")
dt <- aparc_2_mesh(annot = "aparc.a2009s")

## End(Not run)
```

`atlas_lab2ctab` *Label to ctab*

Description

Create a FreeSurfer colortab based on labels. Calls FreeSurfer's `mris_lab2ctab`

Usage

```
atlas_lab2ctab(output_dir, verbose)
```

Arguments

`output_dir`	output directory path
`verbose`	logical indicating to be verbose or not

Value

returns nothing, creates files on the file system

`atlas_vol2label` *Volume to label*

Description

Turn volumetric files into labels for use in annotations. Calls FreeSurfer's `mri_vol2label`.

Usage

```
atlas_vol2label(annot_lab, output_dir, verbose, ncores = 2)
```

Arguments

`annot_lab`	annotation label
`output_dir`	output directory path
`verbose`	logical indicating to be verbose or not
`ncores`	number of cores for parallel processing (default numcores-2)

Value

invisibly returns the list of labels.

`atlas_vol2surf` *Nifti volume to surface*

Description

Transform a Nifti volume to a surface file for FreeSurfer. Calls FreeSurfer's `mri_vol2surf` for the transformation.

Usage

```
atlas_vol2surf(input_file, output_dir, projfrac = 0.5, verbose = TRUE)
```

Arguments

`input_file`	nifti volume
`output_dir`	output directory path
`projfrac`	value to mri_vol2surf -projfrac
`verbose`	logical indicating to be verbose or not

Value

nothing, creates surface files

`change_meshes` *Change meshes to new system*

Description

Change meshes to new system

Usage

```
change_meshes(mesh)
```

Arguments

`mesh`	mesh object

check_fs *Check if FS can be run*

Description

Check if FS can be run

Usage

```
check_fs(msg = NULL)
```

Arguments

msg	message to print on error

Value

logical

curv2ply *Convert Freesurfer curvature file to ply*

Description

Function to convert Freesurfer curvature file into .ply

Usage

```
curv2ply(input_file, output_file = paste(input_file, ".ply"), verbose = TRUE)
```

Arguments

input_file	path to Freesurfer curvature file
output_file	path to output file
verbose	logical indicating to be verbose or not

Value

.ply text

fs_curvatures *Available Freesurfer curvatures*

Description

Available Freesurfer curvatures

Usage

```
fs_curvatures()
```

Value

character

Examples

```
fs_curvatures()
```

fs_nofixcurv *Available Freesurfer no fix curvatures*

Description

Available Freesurfer no fix curvatures

Usage

```
fs_nofixcurv()
```

Value

character

Examples

```
fs_nofixcurv()
```

fs_surfaces *Available Freesurfer surfaces*

Description

Available Freesurfer surfaces

Usage

```
fs_surfaces()
```

Value

character

Examples

```
fs_surfaces()
```

get_mesh *Extract mesh data from ply*

Description

.ply files contain a lot of information. for ggseg3d, we only need information on the vertices and faces of the mesh. Thes function opens a ply file, and organises the meshes and faces into a single list.

Usage

```
get_mesh(ply, ...)
```

Arguments

ply	path to ply-file
...	arguments to read.ply

Value

list of meshes and faces

Examples

```
## Not run:
get_mesh("path/to/surface.ply")

# Turn off showing the ply when reading
get_mesh("path/to/surface.ply", ShowSpecimen = FALSE)

## End(Not run)
```

`ggseg_atlas_repos` *List all online repositories with ggseg-atlases*

Description

Function to easily find all online repositories that contain ggseg atlases

Usage

```
ggseg_atlas_repos(pattern = NULL, ...)
```

Arguments

`pattern`	string pattern to search repos
`...`	additional arguments to `grep`

Value

data frame of online repositories with ggseg-atlases

Examples

```
ggseg_atlas_repos()

ggseg_atlas_repos("yeo")
```

`install_ggseg_atlas` *Install ggseg-atlas from repo*

Description

installs ggseg-atlas library from the ggseg r-universe <https://ggseg.r-universe.dev/ui#builds> .

Usage

```
install_ggseg_atlas(
  package,
  repos = c(ggseg = "https://ggseg.r-universe.dev", getOption("repos")),
  ...
)
```

Arguments

package	package name
repos	vector of repositories to install from. Defaults to ggseg r-universe and CRAN.
...	additional arguments to `install.packages`.

Details

To install, will temporarily alter your install repo settings to also use the ggseg r-universe build as source for packages. These settings will be restored when the function exits.

Examples

```
## Not run:
yeo2011_repo <- ggseg_atlas_repos("yeo2011")
install_ggseg_atlas(yeo2011_repo$repo, yeo2011_repo$source)

## End(Not run)
```

---

install_ggseg_atlas_all

Install all ggseg-atlases registeres

Description

Calls ggseg_atlas_repos, and installs all libraries listed. Be careful calling this function, this will likely take quite some time to complete, and also will contain a lot of heavy data. We recommend only installing atlases you are likely to use and on demand.

Usage

```
install_ggseg_atlas_all(
  repos = c(ggseg = "https://ggseg.r-universe.dev", getOption("repos")),
  ...
)
```

Arguments

repos	repositories to install from. Defaults to ggseg r-universe and a CRAN mirror.
...	additional arguments to `install.packages`.

Examples

```
## Not run:
install_ggseg_atlas_all()

## End(Not run)
```

`is_ctab` *Check if object is colourtable*

Description

Check if object is colourtable

Usage

```
is_ctab(colourtable)
```

Arguments

`colourtable`	data frame with colour table

Value

logical

`lcbc_surf2surf` *Convert LCBC surface file to other subjects*

Description

Convert LCBC surface file to other subjects

Usage

```
lcbc_surf2surf(
  input_volume,
  source_subject = "fsaverage",
  target_subject = "fsaverage5",
  hemisphere = "rh",
  subjects_dir = fs_subj_dir(),
  output_dir = file.path(subjects_dir, target_subject, "surf"),
  cortex = TRUE,
  verbose = TRUE
)
```

Arguments

input_volume	path to input volume
source_subject	source subject
target_subject	target subject
hemisphere	hemisphere, either "rh" or "lh"
subjects_dir	Freesurfer subject directory
output_dir	output directory path
cortex	toggle "–cortex" (TRUE) or "–no-cortex" (FALSE)
verbose	logical indicating to be verbose or not

---

make_aparc_2_3datlas *Create cortical ggseg3d-atlas from annot-file*

Description

Function loops through hemispheres and surfaces to create a data frame that is ready to use with ggseg3d.

Usage

```
make_aparc_2_3datlas(
  annot,
  subject = "fsaverage5",
  hemisphere = c("rh", "lh"),
  surface = c("inflated", "LCBC"),
  subjects_dir = freesurfer::fs_subj_dir(),
  annot_dir = file.path(subjects_dir, subject, "label"),
  output_dir = tempdir(),
  ncores = parallel::detectCores() - 2,
  cleanup = TRUE,
  verbose = TRUE
)
```

Arguments

annot	annotation file, with name [hemi].[annot].annot abd be in annot_dir
subject	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
hemisphere	hemisphere, either "rh" or "lh"
surface	Freesurfer surface
subjects_dir	Freesurfer subject directory
annot_dir	path to directory with annot-files
output_dir	output directory path

ncores	number of cores for parallel processing (default numcores-2)
cleanup	logical to toggle removal of all intermediary files
verbose	logical indicating to be verbose or not

Details

It is recommended that you change the region names for the atlas, and the atlas name to something shorter. See the dk_3d atlas for examples.

Value

nested data frame as ggseg3d-atlas

Examples

```
## Not run:
dt <- aparc_2_3datlas(annot = "aparc.a2009s")
dt <- aparc_2_3datlas(annot = "aparc.a2009s",
                      surface = "sphere")

## End(Not run)
```

---

make_ggseg3d_2_ggseg *Turn ggseg3d-atlas to ggseg*

Description

Function will create a data.frame based on a ggseg3d atlas, based on the contours of each segment.

Usage

```
make_ggseg3d_2_ggseg(
  ggseg3d_atlas,
  steps = 1:7,
  output_dir = tempdir(),
  tolerance = 0,
  smoothness = 5,
  cleanup = FALSE,
  ncores = 2
)
```

Arguments

ggseg3d_atlas	object of class ggseg3d-atlas
steps	numeric vector of steps to run
output_dir	output directory path
tolerance	tolerance during vertex reduction st_simplify

smoothness	smoothing factor, argument to smooth
cleanup	logical to toggle removal of all intermediary files
ncores	number of cores for parallel processing (default numcores-2)

Value

data.frame ready for manual cleaning before turning into a proper ggseg3d-atlas

Examples

```
## Not run:

# Create the DKT atlas as found in the FreeSurfer Subjects directory
# And output the temporary files to the Desktop.
dkt_3d <- make_aparc_2_3datlas(annot = "aparc.DKTatlas",
    output_dir = "~/Desktop/")

## End(Not run)
```

---

make_palette_ggseg *Create ggseg palette from ggseg3d-atlas*

Description

atlases in ggseg have palettes based on colours from the paper originally introducing the atlas. These colours are hard-coded into ggseg3d-atlases. This function extracts those and makes a object ready for incorporation to a ggseg-atlas repository

Usage

```
make_palette_ggseg(ggseg3d_atlas)
```

Arguments

ggseg3d_atlas	ggseg3d-atlas

Value

list with a colour palette

Examples

```
make_palette_ggseg(dk_3d)
```

make_volumetric_2_3datlas

Volumetric segmentation to 3d-atlas

Description

Function to create a ggseg3d-atlas from a volumetric parcellation. Currently only tested using `.mgz` extension images. Will call command-line tools to complete the process.

Usage

```
make_volumetric_2_3datlas(
  template,
  color_lut,
  subject = "fsaverage5",
  subjects_dir = freesurfer::fs_subj_dir(),
  steps = 1:5,
  output_dir = tempdir(),
  verbose = TRUE,
  ncores = parallel::detectCores() - 2,
  cleanup = TRUE
)
```

Arguments

`template`	template `volume.mgz` file path
`color_lut`	Freesurfer colour look-up-table. Either as a path or a data.frame that `is_ctab`
`subject`	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
`subjects_dir`	Freesurfer subject directory
`steps`	if cleanup is disabled, all files are saved, and steps can be re-run individually
`output_dir`	output directory path
`verbose`	logical indicating to be verbose or not
`ncores`	number of cores for parallel processing (default numcores-2)
`cleanup`	logical to toggle removal of all intermediary files

Details

the function is aggressive in the number of cores used to minimize time taken to create the atlas. Set the `ncores` argument for less aggressive approach.

Value

returns a ggseg3d-atlas ready object. Might need manual cleaning to become a good atlas.

Examples

```
## Not run:

fs_subject_dir <- freesurfer::fs_dir()
aseg_temp <- file.path(fs_subject_dir, "fsaverage5/mri/aseg.mgz")
colorlut <- file.path(fs_subject_dir, "ASegStatsLUT.txt")

make_volumetric_2_3datlas(aseg_temp, colorlut)

## End(Not run)
```

`make_volumetric_ggseg` *Make ggseg atlas from volumetric template*

Description

If making an atlas from a non-cortical atlas, volumetric atlases are the best options. Instead of snapshotting images of inflated brain, will snapshot brain slices given x, y, z coordinates for the slices trhoush the `slices` argument.

Usage

```
make_volumetric_ggseg(
  label_file,
  subject = "fsaverage5",
  subjects_dir = fs_subj_dir(),
  output_dir = tempdir(),
  color_lut = NULL,
  steps = 1:8,
  skip_existing = TRUE,
 slices = data.frame(x = c(130, 122, 122), y = c(130, 235, 112), z = c(130, 100, 106),
    view = c("axial", "sagittal", "coronal"), stringsAsFactors = FALSE),
  vertex_size_limits = NULL,
  dilate = NULL,
  tolerance = 0,
  ncores = 2,
  smoothness = 5,
  verbose = TRUE,
  cleanup = FALSE
)
```

Arguments

`label_file`	a volumetric image containing the labels
`subject`	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
`subjects_dir`	Freesurfer subject directory

output_dir	output directory path
color_lut	a file containing color information for the labels
steps	numeric vector of steps to run
skip_existing	logical. If slice snapshots already exist, should these be skipped.
slices	a data.frame with columns x, y, z, and view specifying coordinates and view of slice snapshots.
vertex_size_limits	numeric vector of two, setting the minimum and maximum vector size of polygons. Defaults to NULL, which sets no limits.
dilate	numeric. Dilation factor for polygons. Default NULL applies no dilation.
tolerance	tolerance during vertex reduction st_simplify
ncores	number of cores for parallel processing (default numcores-2)
smoothness	smoothing factor, argument to smooth
verbose	logical indicating to be verbose or not
cleanup	logical to toggle removal of all intermediary files

Value

brain-atlas class

Examples

```
## Not run:

   label_file <- file.path(fs_subj_dir(), subject, "mri/aseg.mgz")
   slices = data.frame(x=130, y=130, z=130, view="axial", stringsAsFactors = FALSE)

   aseg2 <- make_volumetric_ggseg(
      label_file =  label_file,
      slices = slices
   )

   # Have a look at the atlas
   plot(aseg2)

## End(Not run)
```

---

mris_ca_label *MRI ca label*

Description

For a single subject, produces an annotation file, in which each cortical surface vertex is assigned a neuroanatomical label. This automatic procedure employs data from a previously-prepared atlas file. An atlas file is created from a training set, capturing region data manually drawn by neuroanatomists combined with statistics on variability correlated to geometric information derived from the cortical model (sulcus and curvature). Besides the atlases provided with FreeSurfer, new ones can be prepared using mris_ca_train).

Usage

```
mris_ca_label(
  subject = "fsaverage5",
  hemisphere = "lh",
  canonsurf = "sphere.reg",
  classifier = file.path(fs_dir(), "average/lh.DKTatlas40.gcs"),
  output_file,
  subjects_dir = fs_subj_dir(),
  opts = NULL
)
```

Arguments

subject	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
hemisphere	hemisphere, either "rh" or "lh"
canonsurf	canonical surface file. Ie: the name of the spherical surface file which describes the registration of a subject's vertices to the reference "average" surface. Example: sphere.reg
classifier	specify classifier array input file (atlas file)
output_file	path to output file
subjects_dir	Freesurfer subject directory
opts	other arguments to freesurfer command

Examples

```
if(freesurfer::have_fs()){
# for freesurfer help see:
freesurfer::fs_help("mris_ca_label")
mris_ca_label(output_file = "test.lh.annot")

mris_ca_label(hemisphere = "rh", output_file = "test.rh.annot")
}
```

---

mris_label2annot *Convert Label to Annotation*

Description

If you have labels rather than a full annotation file, these can be combined with FreeSurfer's `mris_label2annot`.

Usage

```
mris_label2annot(
  labels,
  hemisphere = "rh",
  ctab,
  subject = "fsaverage5",
  subjects_dir = fs_subj_dir(),
  output_dir = subjects_dir,
  opts = NULL,
  verbose = TRUE
)
```

Arguments

labels	label file path vector
hemisphere	hemisphere, either "rh" or "lh"
ctab	colourtable file
subject	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
subjects_dir	Freesurfer subject directory
output_dir	output directory path
opts	other arguments to freesurfer command
verbose	logical indicating to be verbose or not

Examples

```
if(freesurfer::have_fs()){
# for freesurfer help see:
freesurfer::fs_help("mris_label2annot")

subj_dir <- freesurfer::fs_subj_dir()
# Split up aparc annot into labels
mri_annotation2label(annot_name = "aparc")

# get annot for colour labels
annot <- freesurfer::read_annotation(
              file.path(subj_dir,
                        "fsaverage5/label/rh.aparc.annot"))

labels <- list.files(
   file.path(subj_dir, "fsaverage5/label/aparc"),
    full.names = TRUE)

# Combine them again into annot.
mris_label2annot(labels, annot$colortable)

}
```

`mri_annotation2label` *Convert annotation to label*

Description

Calls FreeSurfer's `mri_annotation2label` to split an annotation file into several labels.

Usage

```
mri_annotation2label(
  annot_name,
  subject = "fsaverage5",
  hemisphere = "lh",
  output_dir = fs_subj_dir(),
  verbose = TRUE,
  opts = NULL
)
```

Arguments

`annot_name`	annotation name. File should exist in subjects label directory
`subject`	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
`hemisphere`	hemisphere, either "rh" or "lh"
`output_dir`	output directory path
`verbose`	logical indicating to be verbose or not
`opts`	other arguments to freesurfer command

Value

nothing. Runs command line to write label files

Examples

```
if(freesurfer::have_fs()){
# for freesurfer help see:
freesurfer::fs_help("mri_annotation2label")
mri_annotation2label(annot_name = "aparc")

mri_annotation2label(annot_name = "aparc.a2009s")

mri_annotation2label(subject = "fsaverage", annot_name = "aparc.a2009s")
}
```

`mri_surf2surf_rereg` *Re-register an annotation file*

Description

Annotation files are subject specific. Most are registered for fsaverage, but we recommend using fsaverage5 for the mesh plots in ggseg3d, as these contain a decent balance in number of vertices for detailed rendering and speed.

Usage

```
mri_surf2surf_rereg(
  subject,
  annot,
  hemi = c("lh", "rh"),
  target_subject = "fsaverage5",
  output_dir = file.path(fs_subj_dir(), subject, "label"),
  verbose = TRUE
)
```

Arguments

`subject`	subject the original annotation file is registered to
`annot`	annotation file name (as found in subjects_dir)
`hemi`	hemisphere (one of "lh" or "rh")
`target_subject`	subject to re-register the annotation (default fsaverage5)
`output_dir`	output directory path
`verbose`	logical indicating to be verbose or not

Value

nothing

Examples

```
if(freesurfer::have_fs()){

# For help see:
freesurfer::fs_help("mri_surf2surf")

mri_surf2surf_rereg(subject = "bert",
                    annot = "aparc.DKTatlas",
                    target_subject = "fsaverage5")
}
```

`mri_vol2label` *Convert volume to label*

Description

Converts values in a volume or surface overlay to a label. The program searches the input for values equal to labelid. The xyz values for each point are then computed based on the tkregister voxel-to-RAS matrix (volume) or from the xyz of the specified surface. The xyz values are then stored in labelfile in the label file format. The statistic value is set to 0. While this program can be used with any mri volume, it was designed to convert parcellation volumes, which happen to be stored in mri format. Calls FreeSurfer's `mri_vol2label`.

Usage

```
mri_vol2label(
  input_file,
  label_id,
  hemisphere,
  output_dir,
  surface = NULL,
  subject = "fsaverage5",
  subjects_dir = fs_subj_dir(),
  opts = NULL,
  verbose = TRUE
)
```

Arguments

`input_file`	input volume
`label_id`	label to run
`hemisphere`	hemisphere, either "rh" or "lh"
`output_dir`	output directory path
`surface`	output surface
`subject`	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
`subjects_dir`	Freesurfer subject directory
`opts`	other arguments to freesurfer command
`verbose`	logical indicating to be verbose or not

Value

returns nothing. Writes a label file.

Examples

```
if(freesurfer::have_fs()){
# for freesurfer help see:
freesurfer::fs_help("mri_vol2label")

out_dir <- tempdir()
vol <- file.path(freesurfer::fs_subj_dir(),
                 "fsaverage5/mri/aseg.mgz")

mri_vol2label(vol,
     label_id = 2,
     hemisphere = "rh",
     output_dir = out_dir)

 # delete output dir when not needed
 unlink(out_dir)
}
```

---

read_ctab *Read colourtab*

Description

Read in a FreeSurfer colortab file.

Usage

```
read_ctab(path)
```

Arguments

path	path to read from

Value

a data.frame with index, label name and RGBA colours

---

restruct_old_3datlas *Change old atlas setup to new*

Description

Change old atlas setup to new

Usage

```
restruct_old_3datlas(atlas_data)
```

Arguments

atlas_data	ggseg3d-atlas object

---

smooth2srf *Turn smooth file to ascii*

Description

Turn smooth file to ascii

Usage

```
smooth2srf(input_file, output_file, verbose)
```

Arguments

input_file	input file path
output_file	path to output file
verbose	logical indicating to be verbose or not

---

subject_2_ascii *Convert subject surface files to ascii*

Description

This function goes through all specified subject, per hemisphere, surface, curvature and no fix curvature specified and turns them into ascii files.

Usage

```
subject_2_ascii(
  subject = "fsaverage5",
  hemisphere = c("rh", "lh"),
  surfaces = fs_surfaces(),
  curvatures = fs_curvatures(),
  nofix_curv = fs_nofixcurv(),
  subjects_dir = freesurfer::fs_subj_dir(),
  output_dir = subjects_dir,
  verbose = TRUE
)
```

Arguments

subject	Freesurfer subject, must exist in whatever subject directory specified or set in the environment with $SUBJECTS_DIR
hemisphere	hemisphere, either "rh" or "lh"
surfaces	string vector of surfaces
curvatures	string vector of curvatures
nofix_curv	string vector of nofix curvatures
subjects_dir	Freesurfer subject directory
output_dir	output directory path
verbose	logical indicating to be verbose or not

Value

no return. Writes files.

Examples

```
## Not run:
subject_2_ascii()
subject_2_ascii("fsaverage")
subject_2_ascii("bert")

## End(Not run)
```

---

surf2ply *Convert Freesurfer surface file into ply*

Description

Function to convert Freesurfer surface file into .ply

Usage

```
surf2ply(input_file, output_file = paste(input_file, ".ply"), verbose = TRUE)
```

Arguments

input_file	path to Freesurfer surface file
output_file	path to output file
verbose	logical indicating to be verbose or not

Value

ply text

surfsplit *Split surface file into separate plys*

Description

Function to split up a surface ply into several ply's based on a freesurfer label file.

Usage

```
surfsplit(
  srf_ply,
  label_path,
  prefix = "test",
  output_dir = dirname(label_path),
  verbose = TRUE
)
```

Arguments

srf_ply	Surface ply path
label_path	label dpv file
prefix	output prefix
output_dir	output directory path
verbose	logical indicating to be verbose or not

Details

Script adapted to R and ggseg based on scripts from Anderson M. Winkler: http://brainder.org

Value

list of ply meshes. Will also write ply meshes to files.

write_ctab *Write colourtab*

Description

write a colortab to file, will be in a format that is accessible by FreeSurfer.

Usage

```
write_ctab(x, path)
```

Arguments

x	colourtab data
path	path to write to

Value

returns nothing, writes file

Index