Package 'ggseg.formats'

April 3, 2026

Title Brain Atlas Data Structures for the 'ggseg' Ecosystem

Version 0.0.2

Description Provides the 'ggseg_atlas' S3 class used across the 'ggseg' ecosystem for 2D and 3D brain visualisation. Ships three bundled atlases ('Desikan-Killiany', 'FreeSurfer' 'aseg', 'TRACULA') and functions for querying, subsetting, renaming, and enriching atlas objects. Also includes readers for 'FreeSurfer' statistics files.

License MIT + file LICENSE

Depends R (>= 4.1.0)

Imports cli, dplyr, ggplot2, lifecycle, rlang, sf, tidyr

Suggests covr, devtools, here, knitr, rmarkdown, spelling, testthat (>= 3.0.0), vdiffr

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website ggsegverse/ggseg.docs

Encoding UTF-8

RoxygenNote 7.3.3

URL https://github.com/ggsegverse/ggseg.formats, https://ggsegverse.github.io/ggseg.formats/

BugReports https://github.com/ggsegverse/ggseg.formats/issues

NeedsCompilation no

Author Athanasia Mo Mowinckel [aut, cre] (ORCID: <https://orcid.org/0000-0002-5756-0223>), Center for Lifespan Changes in Brain and Cognition (LCBC), University of Oslo [cph]

Maintainer Athanasia Mo Mowinckel <a.m.mowinckel@psykologi.uio.no>

Repository CRAN

Date/Publication 2026-04-03 05:10:17 UTC

Contents

`aseg` *FreeSurfer Automatic Subcortical Segmentation Atlas*

Description

Returns the FreeSurfer automatic subcortical segmentation (aseg) atlas containing deep brain structures including the thalamus, caudate, putamen, pallidum, hippocampus, amygdala, accumbens, and ventricles.

Usage

```
aseg()
```

Details

This atlas is derived from FreeSurfer's `aseg.mgz` volumetric segmentation. It works with both ggseg (2D slice views) and ggseg3d (3D mesh visualizations) from a single object.

Value

A `ggseg_atlas` object with components:

atlas Character. Atlas name ("aseg")

type Character. Atlas type ("subcortical")

palette Named character vector of colours for each region

data A `ggseg_data_subcortical` object containing:

- **meshes** Data frame with `label` and `mesh` columns
- **sf** Simple features data frame for 2D rendering

Structures

The atlas contains bilateral structures:

- Thalamus
- Caudate
- Putamen
- Pallidum (globus pallidus)
- Hippocampus
- Amygdala
- Accumbens (nucleus accumbens)
- Ventral diencephalon

Plus midline and ventricular structures:

- Lateral ventricles
- Third ventricle
- Fourth ventricle
- Brain stem
- Cerebellar cortex
- Cerebellar white matter

References

Fischl B, Salat DH, Busa E, et al. (2002). Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. Neuron, 33(3):341-355. doi:10.1016/S08966273(02)00569-X

See Also

`dk()` for cortical parcellation, `ggseg_atlas()` for the atlas class constructor

Other ggseg_atlases: `dk()`, `tracula()`

Examples

```
aseg()
plot(aseg())
atlas_regions(aseg())
```

as_ggseg_atlas *Coerce to ggseg atlas*

Description

Coerce to ggseg atlas

Usage

```
as_ggseg_atlas(x)

as_brain_atlas(x)
```

Arguments

x	object to make into a ggseg_atlas

Value

an object of class 'ggseg_atlas'

Examples

```
atlas <- as_ggseg_atlas(dk())
is_ggseg_atlas(atlas)
```

atlas_labels *Extract unique labels from an atlas*

Description

Extract unique labels from an atlas

Usage

```
atlas_labels(x)

brain_labels(x)
```

Arguments

x	brain atlas

Value

Character vector of atlas region labels

Examples

```
atlas_labels(dk())
atlas_labels(aseg())
```

`atlas_manipulation` *Manipulate brain atlas regions and views*

Description

Functions for modifying brain atlas objects. These cover three areas:

Usage

```
atlas_region_remove(atlas, pattern, match_on = c("region", "label"))

atlas_region_contextual(atlas, pattern, match_on = c("region", "label"))

atlas_region_rename(atlas, pattern, replacement)

atlas_region_keep(atlas, pattern, match_on = c("region", "label"))

atlas_core_add(atlas, data, by = "region")

atlas_view_remove(atlas, views)

atlas_view_keep(atlas, views)

atlas_view_remove_region(
  atlas,
  pattern,
  match_on = c("label", "region"),
  views = NULL
)

atlas_view_remove_small(atlas, min_area, views = NULL)

atlas_view_gather(atlas, gap = 0.15)

atlas_view_reorder(atlas, order, gap = 0.15)
```

Arguments

`atlas`	A ggseg_atlas object
`pattern`	Character pattern to match. Uses `grepl(..., ignore.case = TRUE)`.
`match_on`	Column to match against: `"region"` or `"label"`.

replacement	For atlas_region_rename(): replacement string or function.
data	For atlas_core_add(): data.frame with metadata to join.
by	For atlas_core_add(): column(s) to join by. Default "region".
views	For view functions: character vector of view names or patterns. Multiple values collapsed with "|" for matching.
min_area	For atlas_view_remove_small(): minimum polygon area to keep. Context geometries are never removed.
gap	Proportional gap between views (default 0.15 = 15% of max width).
order	For atlas_view_reorder(): character vector of desired view order. Unspecified views appended at end.

Details

Region manipulation modifies which regions are active in the atlas:

- atlas_region_remove(): completely remove regions
- atlas_region_contextual(): keep geometry but remove from core/palette
- atlas_region_rename(): rename regions in core
- atlas_region_keep(): keep only matching regions

View manipulation modifies the 2D sf geometry data:

- atlas_view_remove(): remove entire views
- atlas_view_keep(): keep only matching views
- atlas_view_remove_region(): remove specific region geometry from sf
- atlas_view_remove_small(): remove small polygon fragments
- atlas_view_gather(): reposition views to close gaps
- atlas_view_reorder(): change view order

Core manipulation modifies atlas metadata:

- atlas_core_add(): join additional metadata columns

Value

Modified ggseg_atlas object

Functions

- atlas_region_contextual(): Keep geometry for visual context but remove from core, palette, and 3D data. Context geometries render grey and don't appear in legends.
- atlas_region_rename(): Rename regions matching a pattern. Only affects the region column, not label. If replacement is a function, it receives matched names and returns new names.
- atlas_region_keep(): Keep only matching regions. Non-matching regions are removed from core, palette, and 3D data but sf geometry is preserved for surface continuity.

- `atlas_core_add()`: Join additional metadata columns to atlas core.
- `atlas_view_remove()`: Remove views matching pattern from sf data.
- `atlas_view_keep()`: Keep only views matching pattern.
- `atlas_view_remove_region()`: Remove specific region geometry from sf data only. Core, palette, and 3D data are unchanged.
- `atlas_view_remove_small()`: Remove region geometries below a minimum area threshold. Context geometries (labels not in core) are never removed. Optionally scope to specific views.
- `atlas_view_gather()`: Reposition remaining views to close gaps after view removal.
- `atlas_view_reorder()`: Reorder views and reposition. Views not in `order` are appended at end.

Examples

```
dk() |>
  atlas_region_remove("bankssts") |>
  atlas_region_keep("frontal", match_on = "region")
```

---

atlas_meshes *Get atlas meshes for 3D rendering*

Description

Returns meshes data joined with core region info and palette colours. Used for subcortical and tract atlases.

Usage

```
atlas_meshes(atlas)
```

Arguments

`atlas`	a ggseg_atlas object

Value

data.frame with meshes ready for 3D rendering

Examples

```
meshes <- atlas_meshes(aseg())
head(meshes)
```

atlas_palette *Get atlas palette*

Description

Retrieves the colour palette from a brain atlas.

Usage

```
atlas_palette(name = "dk", ...)
```

Arguments

name	Character name of atlas (e.g., "dk", "aseg") or a ggseg_atlas object
...	Additional arguments (unused)

Value

Named character vector of colours

Examples

```
atlas_palette(aseg())
atlas_palette(dk())
```

atlas_regions *Extract unique region names from an atlas*

Description

Extract unique region names from an atlas

Usage

```
atlas_regions(x)

brain_regions(x)
```

Arguments

x	brain atlas

Value

Character vector of region names

Examples

```
atlas_regions(dk())
atlas_regions(aseg())
```

`atlas_sf` *Get atlas data for 2D rendering*

Description

Returns sf data joined with core region info and palette colours.

Usage

```
atlas_sf(atlas)
```

Arguments

`atlas`	a ggseg_atlas object

Value

sf data.frame ready for plotting

Examples

```
sf_data <- atlas_sf(dk())
head(sf_data)
```

`atlas_type` *Detect atlas type*

Description

Detect atlas type

Usage

```
atlas_type(x)
```

Arguments

`x`	brain atlas object

Value

Character string: "cortical", "subcortical", or "tract"

Examples

```
atlas_type(dk())
atlas_type(aseg())
atlas_type(tracula())
```

`atlas_vertices` *Get atlas vertices for 3D rendering*

Description

Returns vertices data joined with core region info and palette colours. Used for cortical atlases with vertex-based rendering.

Usage

```
atlas_vertices(atlas)
```

Arguments

`atlas`	a ggseg_atlas object

Value

data.frame with vertices ready for 3D rendering

Examples

```
verts <- atlas_vertices(dk())
head(verts)
```

`atlas_views` *Get available views in atlas*

Description

Get available views in atlas

Usage

```
atlas_views(atlas)

brain_views(atlas)
```

Arguments

`atlas`	A ggseg_atlas object

Value

Character vector of view names, or NULL if no sf data

Examples

```
atlas_views(aseg())
atlas_views(tracula())
```

---

convert_legacy_brain_atlas

Convert legacy ggseg atlases to ggseg_atlas format

Description

[Superseded]

Convert old-style ggseg (2D) and ggseg3d (3D) atlases into the new `ggseg_atlas` format. This is a bridge function for working with existing atlases during the transition period.

For new atlases, use `ggsegExtra::create_cortical_atlas()` or `ggsegExtra::create_subcortical_atlas()` instead - they produce better results with proper vertex indices.

The function handles three scenarios:

- **Both 2D and 3D**: Merges geometry with vertex data
- **3D only**: Extracts vertices, infers indices from mesh coordinates
- **2D only**: Keeps geometry, 3D rendering unavailable

If the 3D atlas already contains vertex indices (newer ggseg3d atlases), those are preserved. Otherwise, vertex indices are inferred from mesh coordinates using hash-based matching (no FreeSurfer needed).

Usage

```
convert_legacy_brain_atlas(
  atlas_2d = NULL,
  atlas_3d = NULL,
  atlas_name = NULL,
  type = NULL,
  surface = "inflated",
  brain_meshes = NULL
)

unify_legacy_atlases(
  atlas_2d = NULL,
  atlas_3d = NULL,
  atlas_name = NULL,
  type = NULL,
```

```
  surface = "inflated",
  brain_meshes = NULL
)
```

Arguments

atlas_2d	A ggseg_atlas (or legacy brain_atlas) with 2D geometry, or NULL.
atlas_3d	A ggseg3d_atlas with mesh data, or NULL.
atlas_name	Name for the output atlas. If NULL, derived from input.
type	Atlas type: "cortical" or "subcortical". If NULL, inferred from the input atlases.
surface	Which surface to match against when inferring vertices (e.g., "inflated"). Must match the 3D atlas surface exactly.
brain_meshes	Optional user-supplied brain meshes for vertex inference.

Value

A ggseg_atlas object.

Examples

```
new_atlas <- convert_legacy_brain_atlas(atlas_2d = dk())
```

---

dk *Desikan-Killiany Cortical Atlas*

Description

Returns the Desikan-Killiany cortical parcellation atlas with 34 regions per hemisphere (68 total) on the cortical surface.

Usage

```
dk()
```

Details

This atlas is based on the FreeSurfer aparc annotation and is one of the most widely used cortical parcellations in neuroimaging research.

The atlas works with both ggseg (2D polygon plots) and ggseg3d (3D mesh visualizations) from a single object.

Value

A `ggseg_atlas` object with components:

atlas Character. Atlas name ("dk")

type Character. Atlas type ("cortical")

palette Named character vector of colours for each region

data A `ggseg_data_cortical` object containing:

vertices Data frame with `label` and `vertices` columns

sf Simple features data frame for 2D rendering

Regions

The atlas contains 34 regions per hemisphere including: banks of superior temporal sulcus, caudal anterior cingulate, caudal middle frontal, cuneus, entorhinal, fusiform, inferior parietal, inferior temporal, isthmus cingulate, lateral occipital, lateral orbitofrontal, lingual, medial orbitofrontal, middle temporal, parahippocampal, paracentral, pars opercularis, pars orbitalis, pars triangularis, pericalcarine, postcentral, posterior cingulate, precentral, precuneus, rostral anterior cingulate, rostral middle frontal, superior frontal, superior parietal, superior temporal, supramarginal, frontal pole, temporal pole, transverse temporal, and insula.

References

Desikan RS, Segonne F, Fischl B, et al. (2006). An automated labeling system for subdividing the human cerebral cortex on MRI scans into gyral based regions of interest. NeuroImage, 31(3):968-980. doi:10.1016/j.neuroimage.2006.01.021

Fischl B, van der Kouwe A, Destrieux C, et al. (2004). Automatically parcellating the human cerebral cortex. Cerebral Cortex, 14(1):11-22. doi:10.1093/cercor/bhg087

See Also

`aseg()` for subcortical structures, `ggseg_atlas()` for the atlas class constructor

Other ggseg_atlases: `aseg()`, `tracula()`

Examples

```
dk()
plot(dk())
atlas_regions(dk())
atlas_labels(dk())
```

get_brain_mesh *Get brain surface mesh*

Description

Retrieves a brain surface mesh for the specified hemisphere and surface type. By default, provides the inflated fsaverage5 surface from internal data. Other surfaces (pial, white, semi-inflated) require the ggseg3d package.

Usage

```
get_brain_mesh(
  hemisphere = c("lh", "rh"),
  surface = "inflated",
  brain_meshes = NULL
)
```

Arguments

`hemisphere`	`"lh"` or `"rh"`
`surface`	Surface type (default `"inflated"`). Other surfaces require ggseg3d or a custom `brain_meshes` argument.
`brain_meshes`	Optional user-supplied mesh data. Accepts either `list(lh = list(vertices, faces), rh = ...)` or the legacy `list(lh_inflated = list(vertices, faces), ...)` format.

Value

A list with `vertices` (data.frame with x, y, z) and `faces` (data.frame with i, j, k), or NULL if the mesh is not available.

Examples

```
mesh <- get_brain_mesh("lh")
head(mesh$vertices)
```

get_cerebellar_mesh *Get SUIT cerebellar surface mesh*

Description

Retrieves the shared SUIT cerebellar surface mesh used for vertex-based cerebellar atlas rendering. The mesh contains 28,935 original surface vertices plus 1,078 additional vertices forming a cap over the peduncular surface (where the cerebellum meets the brainstem). The cap vertices (indices 28,935–30,012) are duplicates of the boundary loop and a centroid, designed to render as an opaque grey wall when not assigned to any atlas region.

Usage

```
get_cerebellar_mesh()
```

Value

A list with `vertices` (data.frame with x, y, z) and `faces` (data.frame with i, j, k, 0-based indices).

Examples

```
mesh <- get_cerebellar_mesh()
nrow(mesh$vertices)
```

`ggseg_atlas` *Constructor for ggseg atlas*

Description

Creates an object of class 'ggseg_atlas' for plotting brain parcellations using ggseg (2D) and ggseg3d (3D).

Usage

```
ggseg_atlas(atlas, type, core, data, palette = NULL)

brain_atlas(atlas, type, core, data, palette = NULL)
```

Arguments

`atlas`	atlas short name, length one
`type`	atlas type: "cortical", "subcortical", "tract", or "cerebellar"
`core`	data.frame with required columns hemi, region, label (one row per unique region). May contain additional columns for grouping or metadata (e.g., lobe, network, Brodmann area).
`data`	a ggseg_atlas_data object created by `ggseg_data_cortical()`, `ggseg_data_subcortical()`, `ggseg_data_tract()`, or `ggseg_data_cerebellar()`. Must match the specified type.
`palette`	named character vector of colours keyed by label

Value

an object of class 'ggseg_atlas'

Examples

```
core <- data.frame(
  hemi = c("left", "left"),
  region = c("region1", "region2"),
  label = c("lh_region1", "lh_region2")
)
vertices <- data.frame(
  label = c("lh_region1", "lh_region2"),
  vertices = I(list(c(1L, 2L, 3L), c(4L, 5L, 6L)))
)
atlas <- ggseg_atlas(
  atlas = "test",
  type = "cortical",
  core = core,
  data = ggseg_data_cortical(vertices = vertices)
)
```

---

ggseg_data_cerebellar *Create cerebellar atlas data*

Description

Creates a data object for cerebellar brain atlases. Cerebellar atlases use sf polygons from a SUIT flatmap for 2D rendering and vertex indices into the shared SUIT cerebellar surface mesh for 3D rendering.

Usage

```
ggseg_data_cerebellar(sf = NULL, vertices = NULL, meshes = NULL)
```

Arguments

sf	sf data.frame with columns label, view, geometry for 2D rendering. Surface regions use view "flatmap"; deep structures use other views.
vertices	data.frame with columns label and vertices (list-column of integer vectors). Each vector contains 0-based vertex indices into the SUIT cerebellar surface (see get_cerebellar_mesh()). Only for surface regions.
meshes	Optional data.frame with columns label and mesh (list-column of mesh objects with vertices and faces). For deep cerebellar structures that are not on the cortical surface. Same format as ggseg_data_subcortical() meshes.

Details

The shared mesh (see get_cerebellar_mesh()) includes a cap over the peduncular surface where the cerebellum meets the brainstem. Vertices on this cap (indices 28,935–30,012) are not assigned to any atlas region and render as na_colour in 3D, analogous to the medial wall in cortical atlases.

Deep cerebellar structures (e.g. dentate, interposed, fastigial nuclei) that are not on the cortical surface are stored as individual per-region meshes in meshes, following the same format as subcortical atlases. Their 2D sf geometries use views other than "flatmap" (e.g. "nuclei").

Value

An object of class c("ggseg_data_cerebellar", "ggseg_atlas_data")

`ggseg_data_cortical` *Create cortical atlas data*

Description

Creates a data object for cortical brain atlases. Cortical atlases use vertex indices that map regions to vertices on a shared brain surface mesh (e.g., fsaverage5).

Usage

```
ggseg_data_cortical(sf = NULL, vertices = NULL)

brain_data_cortical(sf = NULL, vertices = NULL)
```

Arguments

`sf`	sf data.frame with columns label, view, geometry for 2D rendering. Optional but required for ggseg plotting.
`vertices`	data.frame with columns label and vertices (list-column of integer vectors). Each vector contains vertex indices for that region.

Value

An object of class c("ggseg_data_cortical", "ggseg_atlas_data")

Examples

```
data <- ggseg_data_cortical(
  vertices = data.frame(
    label = c("bankssts", "caudalanteriorcingulate"),
    vertices = I(list(c(1L, 2L, 3L), c(4L, 5L, 6L)))
  )
)
```

ggseg_data_subcortical *Create subcortical atlas data*

Description

Creates a data object for subcortical brain atlases. Subcortical atlases use individual 3D meshes for each structure (e.g., hippocampus, amygdala).

Usage

```
ggseg_data_subcortical(sf = NULL, meshes = NULL)

brain_data_subcortical(sf = NULL, meshes = NULL)
```

Arguments

sf	sf data.frame with columns label, view, geometry for 2D rendering. Optional.
meshes	data.frame with columns label and mesh (list-column). Each mesh is a list with:

- vertices: data.frame with x, y, z columns
- faces: data.frame with i, j, k columns (1-based triangle indices)

Value

An object of class c("ggseg_data_subcortical", "ggseg_atlas_data")

Examples

```
data <- ggseg_data_subcortical(
  meshes = data.frame(
    label = "hippocampus_left",
    mesh = I(list(list(
      vertices = data.frame(x = 1:10, y = 1:10, z = 1:10),
      faces = data.frame(i = 1:3, j = 2:4, k = 3:5)
    )))
  )
)
```

`ggseg_data_tract` *Create tract atlas data*

Description

Creates a data object for white matter tract atlases. Stores centerlines compactly; tube meshes are generated at render time for efficiency.

Usage

```
ggseg_data_tract(sf = NULL, centerlines = NULL, meshes = NULL, ...)

brain_data_tract(sf = NULL, centerlines = NULL, meshes = NULL, ...)
```

Arguments

`sf`	sf data.frame with columns label, view, geometry for 2D rendering. Optional.
`centerlines`	data.frame with columns: • label: tract identifier (character) • points: list-column of n x 3 matrices (centerline coordinates) • tangents: list-column of n x 3 matrices (for orientation coloring)
`meshes`	Deprecated. Use centerlines instead. If provided, will be converted to centerlines format.
`...`	Absorbs legacy fields (e.g. tube_radius, tube_segments) from old cached atlas objects.

Value

An object of class c("ggseg_data_tract", "ggseg_atlas_data")

Examples

```
centerlines_df <- data.frame(
  label = "cst_left",
  points = I(list(matrix(rnorm(150), ncol = 3))),
  tangents = I(list(matrix(rnorm(150), ncol = 3)))
)
data <- ggseg_data_tract(centerlines = centerlines_df)
```

is_ggseg3d_atlas *Check if object is a legacy ggseg3d atlas*

Description

Check if object is a legacy ggseg3d atlas

Usage

```
is_ggseg3d_atlas(x)
```

Arguments

x	an object

Value

logical

Examples

```
is_ggseg3d_atlas(dk())
```

is_ggseg_atlas *Check ggseg atlas class*

Description

These functions check both the class tag and structural validity by passing the object through ggseg_atlas(). An object that carries the right class but fails validation returns FALSE.

Usage

```
is_ggseg_atlas(x)

is_cortical_atlas(x)

is_subcortical_atlas(x)

is_tract_atlas(x)

is_cerebellar_atlas(x)

is_brain_atlas(x)
```

Arguments

x	an object

Value

logical

Examples

```
is_ggseg_atlas(dk())
is_cortical_atlas(dk())
is_subcortical_atlas(aseg())
is_tract_atlas(tracula())
```

---

read_atlas_files *Read in atlas data from all subjects*

Description

Recursively reads in all stats files for an atlas (given a unique character string), for all subjects in the subjects directory. Will add hemisphere and subject id to the data.

Usage

```
read_atlas_files(subjects_dir, atlas)
```

Arguments

subjects_dir	FreeSurfer subject directory
atlas	unique character combination identifying the atlas

Value

tibble with stats information for subjects from FreeSurfer

Examples

```
subj_dir <- "/path/to/freesurfer/7.2.0/subjects/"
read_atlas_files(subj_dir, "aseg.stats")
read_atlas_files(subj_dir, "lh.aparc.stats")
```

read_freesurfer_stats *Read in raw FreeSurfer stats file*

Description

FreeSurfer atlas stats files have a format that can be difficult to easily read in to R. This function takes a raw stats-file from the subjects directory and reads it in as a data.frame.

Usage

```
read_freesurfer_stats(path, rename = TRUE)
```

Arguments

path	path to stats file
rename	logical. rename headers for ggseg compatibility

Value

tibble with stats information for subjects from FreeSurfer

Examples

```
subj_dir <- "/path/to/freesurfer/7.2.0/subjects/"
aseg_stats <- file.path(subj_dir, "bert/stats/aseg.stats")
read_freesurfer_stats(aseg_stats)
```

read_freesurfer_table *Read in stats table from FreeSurfer*

Description

FreeSurfer has functions to create tables from raw stats files. If you have data already merged using the `aparcstats2table` or `asegstats2table` from FreeSurfer, this function will read in the data and prepare it for ggseg.

Usage

```
read_freesurfer_table(path, measure = NULL, ...)
```

Arguments

path	path to the table file
measure	which measure is the table of
...	additional arguments to `read.table`

Value

tibble with stats information for subjects from FreeSurfer

Examples

```
file_path <- "all_subj_aseg.txt"
read_freesurfer_table(file_path)
```

`tracula` *TRACULA White Matter Tract Atlas*

Description

Returns the TRACULA (TRActs Constrained by UnderLying Anatomy) white matter bundle atlas in MNI space.

Usage

```
tracula()
```

Details

This atlas contains major white matter tracts reconstructed from diffusion MRI using FreeSurfer's TRACULA training data. It works with both ggseg (2D slice projections) and ggseg3d (3D tube mesh visualizations).

Value

A `ggseg_atlas` object with components:

atlas Character. Atlas name ("tracula")

type Character. Atlas type ("tract")

palette Named character vector of colours for each tract

data A `ggseg_data_tract` object containing:

- **centerlines** List of centerline matrices per tract
- **sf** Simple features data frame for 2D rendering

References

Yendiki A, Panneck P, Srinivasan P, et al. (2011). Automated probabilistic reconstruction of white-matter pathways in health and disease using an atlas of the underlying anatomy. Frontiers in Neuroinformatics, 5:23. doi:10.3389/fninf.2011.00023

See Also

dk() for cortical parcellation, aseg() for subcortical structures, ggseg_atlas() for the atlas class constructor

Other ggseg_atlases: aseg(), dk()

Examples

```
tracula()
plot(tracula())
atlas_regions(tracula())
```

Index