Package 'ciftiTools'

November 20, 2020

Type Package

Title Tools for Reading and Visualizing CIFTI Brain Files

Version 0.1.6.0

Maintainer Amanda Mejia <mandy.mejia@gmail.com>

Description CIFTI files contain brain imaging data in ``gray-ordinates'', which represent the gray matter as cortical surface vertices (left and right) and subcortical voxels (cerebellum, basal ganglia, and other deep gray matter). 'ciftiTools' uses the Connectome Workbench to read CIFTI files into R and apply common pre-processing steps (e.g. smoothing, resampling). It also provides tools for visualizing the cortical surface with GIFTI files, and for visualizing the subcortical volume.

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

LazyData true

Imports gifti (> 0.7.5), grDevices, oro.nifti, RNifti, RColorBrewer, xml2

Suggests fields, ggplot2, ggpubr, knitr, rmarkdown, papayar, rgl, testthat (>= 2.1.0), covr

RoxygenNote 7.1.1

URL https://github.com/mandymejia/ciftiTools

BugReports https://github.com/mandymejia/ciftiTools/issues

NeedsCompilation no

Author Amanda Mejia [aut, cre], Damon Pham [ctb] (<https://orcid.org/0000-0001-7563-4727>)

Repository CRAN

Date/Publication 2020-11-20 10:20:05 UTC

R topics documented:

`add_surf` *Add surface(s) to a* `"xifti"` *object*

Description

Add left or right cortical surface geometry to a `"xifti"` object.

Usage

```
add_surf(xifti, surfL = NULL, surfR = NULL)
```

Arguments

xifti	Object of class "xifti". See is.xifti and make_xifti.
surfL	(Optional) Left brain surface model. Can be a file path to a GIFTI surface geometry file (ends in "*.surf.gii"), a "gifti" object representing surface geometry, or a "surf" object.
surfR	(Optional) Right brain surface model. Can be a file path to a GIFTI surface geometry file (ends in "*.surf.gii"), a "gifti" object representing surface geometry, or a "surf" object.

Details

surfL will be added to xifti$surf$cortex_left and surfR will be added to xifti$surf$cortex_right. Any existing surfaces will be overwritten.

Value

the "xifti" object with added surface geometry components.

---

as.xifti *Assemble a "xifti" object from data*

Description

Assembles cortical data, subcortical data, and/or surface geometry to form a "xifti" object. The inputs must be data objects (vectors, matrices or arrays, depending on the argument). as.xifti can be used to combine the files written by separate_cifti, or read individual components independent of any CIFTI file.

Usage

```
as.xifti(
  cortexL = NULL,
  cortexL_mwall = NULL,
  cortexR = NULL,
  cortexR_mwall = NULL,
  mwall_values = c(NA, NaN),
  subcortVol = NULL,
  subcortLabs = NULL,
  subcortMask = NULL,
  surfL = NULL,
  surfR = NULL,
  col_names = NULL,
  HCP_32k_auto_mwall = TRUE
)

as_xifti(
```

```
  cortexL = NULL,
  cortexL_mwall = NULL,
  cortexR = NULL,
  cortexR_mwall = NULL,
  mwall_values = c(NA, NaN),
  subcortVol = NULL,
  subcortLabs = NULL,
  subcortMask = NULL,
  surfL = NULL,
  surfR = NULL
)

as.cifti(
  cortexL = NULL,
  cortexL_mwall = NULL,
  cortexR = NULL,
  cortexR_mwall = NULL,
  mwall_values = c(NA, NaN),
  subcortVol = NULL,
  subcortLabs = NULL,
  subcortMask = NULL,
  surfL = NULL,
  surfR = NULL
)

as_cifti(
  cortexL = NULL,
  cortexL_mwall = NULL,
  cortexR = NULL,
  cortexR_mwall = NULL,
  mwall_values = c(NA, NaN),
  subcortVol = NULL,
  subcortLabs = NULL,
  subcortMask = NULL,
  surfL = NULL,
  surfR = NULL
)
```

Arguments

`cortexL`, `cortexL_mwall`

Left cortex data and ROI. Each must be a data matrix or vector.

If `cortexL_mwall` is not provided, `cortexL` should have data for all vertices on the left cortical surface ($V_L x T$ data matrix). There will not be a mask for the medial wall. Not providing the medial wall mask is appropriate for ".dlabels.nii" files where the medial wall may have its own label and therefore should not be treated as missing data.

If `cortexL_mwall` is provided, `cortexL` should either have data for all vertices on the left cortical surface ($V_L x T$ data matrix, with filler values e.g. `0`

or NaN for medial wall vertices), or have data only for non-medial wall vertices ($(V_L - mwall_L)xT$ data matrix). The medial wall mask will be the 0 values in cortexL_mwall. The medial wall mask should be provided whenever the medial wall should be treated as missing data.

Since the unmasked cortices must have the same number of vertices, V_L should match V_R.

cortexR, cortexR_mwall

Right cortex data and ROI. Each must be a data matrix or vector.

If cortexR_mwall is not provided, cortexR should have data for all vertices on the right cortical surface (V_RxT data mre will not be a mask for the medial wall. Not providing the medial wall mask is appropriate for ".dlabels.nii" files where the medial wall may have its own label and therefore should not be treated as missing data.

If cortexR_mwall is provided, cortexR should either have data for all vertices on the right cortical surface (V_RxT data matrix, with filler values e.g. 0 or NaN for medial wall vertices), or have data only for non-medial wall vertices ($(V_R - mwall_R)xT$ data matrix). The medial wall mask will be the 0 values in cortexR_mwall. The medial wall mask should be provided whenever the medial wall should be treated as missing data.

Since the unmasked cortices must have the same number of vertices, V_L should match V_R.

mwall_values If cortex[L/R]_mwall was not provided, or if it was invalid (i.e. bad length or all TRUE), the medial wall mask will be inferred from rows in cortex[L/R] that are constantly one of these values. Default: c(NA,NaN). If NULL, do not attempt to infer the medial wall from the data values. NULL should be used if NA or NaN are legitimate values that non-medial wall vertices might take on.

subcortVol, subcortLabs, subcortMask

subcortVol represents the data values of the subcortex. It is either a 3D/4D numeric array ($ixjxkxT$), or a vectorized matrix (V_S voxels by T measurements). If it's vectorized, the voxels should be in spatial order (i index increasing fastest, then j, then k).

subcortLabs represents the brainstructure labels of each voxel: see substructure_table. It is either a 3D data array ($ixjxk$) of integer brainstructure indices, or a V_S length vector in spatial order with brainstructure names as factors or integer indices. The indices should be 3-21 (1 and 2 correspond to left and right cortex, respectively) or 1-19 (cortex labels omitted), with 0 representing out-of-mask voxels.

subcortMask is logical 3D data array ($ixjxk$) where TRUE values indicate subcortical voxels (in-mask). If it is not provided, the mask will be inferred from voxels with labels 0, NA, or NaN in subcortLabs. If subcortLabs are vectorized and subcortMask is not provided, the mask cannot be inferred so an error will occur.

surfL, surfR (Optional) Surface geometries for the left or right cortex. Can be a surface GIFTI file path or "surf" object; see make_surf for a full description of valid inputs.

col_names Names of each measurement/column in the data.

`HCP_32k_auto_mwall`
: If left and/or right cortex data is provided, and the number of vertices matches that of the HCP 32k mesh (29696 on left, and 29716 on right), should the medial wall masks be added to the `"xifti"` if not provided? Default: `TRUE`.

Details

Each data or surface component is optional. Metadata components (`cortex[L/R]_mwall`, `subcortLabs`, and `subcortMask`) will be ignored if its corresponding data component is not provided. If no data or surface components are provided, then the `template_xifti` will be returned.

If cortical data are provided without a corresponding medial wall mask, or if the provided mask is invalid or empty, then the medial wall will be inferred from data rows that are constantly a value in `mwall_values`. But if `mwall_values` is `NULL`, no attempt to infer the medial wall will be made and the medial wall metadata entry will be `NULL`.

The total number of greyordinates will be $G = (V_L - mwall_L) + (V_R - mwall_R) + V_S$: $V_L - mwall_L$ left vertices, $V_R - mwall_R$ right vertices and V_S subcortical voxels. T, the total number of measurements (columns of data), must be the same for each brainstructure.

Value

A `"xifti"` object

`brainstructures_Param_all` *brainstructures*

Description

brainstructures

Arguments

`brainstructures`
: Character vector indicating which brain structure(s) to obtain: `"left"` (left cortical surface), `"right"` (right cortical surface) and/or `"subcortical"` (subcortical and cerebellar gray matter). Can also be `"all"` (obtain all three brain structures). Default: `"all"`.

  If a brain structure is indicated but does not exist, a warning will be raised and that brain structure will be skipped.

brainstructures_Param_LR *brainstructures*

Description

brainstructures

Arguments

`brainstructures`

Character vector indicating which brain structure(s) to obtain: `"left"` (left cortical surface), `"right"` (right cortical surface) and/or `"subcortical"` (subcortical and cerebellar gray matter). Can also be `"all"` (obtain all three brain structures). Default: `c("left","right")` (cortical surface only).

If a brain structure is indicated but does not exist, a warning will be raised and that brain structure will be skipped.

ciftiTools.getOption *Get* `ciftiTools` *option*

Description

Gets an R option (with prefix "ciftiTools_") value. See `ciftiTools.listOptions`.

Usage

```
ciftiTools.getOption(opt)
```

Arguments

`opt`	The option.

Value

The value, `val`

ciftiTools.listOptions *List* ciftiTools *options*

Description

List ciftiTools options

Usage

```
ciftiTools.listOptions()
```

Value

data.frame describing the options

ciftiTools.setOption *Set* ciftiTools *option*

Description

Sets an R option (with prefix "ciftiTools_"). See ciftiTools.listOptions.

Usage

```
ciftiTools.setOption(opt, val)
```

Arguments

opt	The option.
val	The value to set the option as.

Value

The new value, val

cifti_fname_Param *cifti_fname*

Description

cifti_fname

Arguments

cifti_fname	File path of CIFTI-format data (ending in ".d*.nii").

Connectome_Workbench_Description *Connectome_Workbench*

Description

Connectome_Workbench

Connectome Workbench Requirement

This function uses a system wrapper for the 'wb_command' executable. The user must first download and install the Connectome Workbench, available from https://www.humanconnectome.org/software/get-connectome-workbench . The `wb_path` argument is the full file path to the Connectome Workbench folder. (The full file path to the 'wb_cmd' executable also works.)

demo_files *Get example files*

Description

Get the file paths of the example CIFTI and surface GIFTI files included with `ciftiTools`.

Usage

```
demo_files()
```

Details

These files are from NITRC: cifti-2_test_data-1.2.zip at https://www.nitrc.org/frs/?group_id=454

Value

A list with CIFTI file names in the first entry, and surface file names in the second.

expand_color_pal *Interpolates between entries in the input palette to make a larger palette with at least MIN_COLOR_RES entries.*

Description

Interpolates between entries in the input palette to make a larger palette with at least MIN_COLOR_RES entries.

Usage

```
expand_color_pal(pal, MIN_COLOR_RES = 255)
```

Arguments

pal	The color palette to expand, as a data.frame with two columns: "color" (character: color hex codes) and "value" (numeric).
MIN_COLOR_RES	The minimum number of entries to have in the output palette. Because of rounding, there may be more than this number of entries.

Value

A data.frame with two columns: "color" (character: color hex codes) and "value" (numeric)

get_wb_cmd_path *Get the Connectome Workbench command path*

Description

Retrieves the Connectome Workbench path from the ciftiTools_wb_path option or the provided argument. This path can be the Connectome Workbench folder or the 'wb_command' executable.

Usage

```
get_wb_cmd_path(wb_path = NULL, verbose = FALSE)
```

Arguments

wb_path	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with ciftiTools.setOption("wb_path","path/to/workbench").
verbose	Default: FALSE.

`info_cifti` *Get CIFTI metadata*

Description

Get CIFTI metadata from the NIFTI header and XML using the Connectome Workbench command `-nifti-information`. The information is formatted as the `meta` component in a `"xifti"` object (see `template_xifti`), and includes:

1. medial wall masks for the left and right cortex
2. the subcortical labels (ordered spatially)
3. the subcortical mask
4. other NIFTI intent-specific metadata

Usage

```
info_cifti(cifti_fname, wb_path = NULL)

infoCIfTI(cifti_fname, wb_path = NULL)

infocii(cifti_fname, wb_path = NULL)
```

Arguments

`cifti_fname`	File path of CIFTI-format data (ending in ".d*.nii").
`wb_path`	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with `ciftiTools.setOption("wb_path","path/to/workbench")`.

Details

Additional metadata depends on the type of CIFTI file:

1. "dtseries"
   (a) time_start Start time
   (b) time_step The TR
   (c) time_unit Unit of time
2. "dscalar"
   (a) names Name of each data column
3. "dlabels"
   (a) names(Names of each data column.)
   (b) labels(List of $Lx5$ data.frames. Row names are the label names. Column names are Key, Red, Green, Blue, and Alpha. List entry names are the names of each data column.)

Value

The metadata component of a `"xifti"` for the input CIFTI file

Label Levels

`xifti$meta$subcort$labels` is a factor with the following levels:

1. Cortex-L
2. Cortex-R
3. Accumbens-L
4. Accumbens-R
5. Amygdala-L
6. Amygdala-R
7. Brain Stem
8. Caudate-L
9. Caudate-R
10. Cerebellum-L
11. Cerebellum-R
12. Diencephalon-L
13. Diencephalon-R
14. Hippocampus-L
15. Hippocampus-R
16. Pallidum-L
17. Pallidum-R
18. Putamen-L
19. Putamen-R
20. Thalamus-L
21. Thalamus-R

These correspond to the same structures as given by `ft_read_cifti` in the `cifti-matlab` MATLAB toolbox.

Connectome Workbench Requirement

This function uses a system wrapper for the 'wb_command' executable. The user must first download and install the Connectome Workbench, available from https://www.humanconnectome.org/software/get-connectome-workbench . The `wb_path` argument is the full file path to the Connectome Workbench folder. (The full file path to the 'wb_cmd' executable also works.)

`is.cifti` *Validate a "xifti" object*

Description

Check if object is valid for a "xifti" object. This alias for `is.xifti` is offered as a convenience, and a message will warn the user. We recommend using `is.xifti` instead.

Usage

```
is.cifti(x, messages = TRUE)

is_cifti(x, messages = TRUE)

isCIfTI(x, messages = TRUE)
```

Arguments

x	The putative "xifti" object.
messages	If x is not a "xifti" object, print messages explaining the problem? Default is TRUE.

Details

Requirements: it is a list with the same structure as `template_xifti`. The size of each data entry must be compatible with its corresponding mask (medial wall for the cortex and volumetric mask for the subcortex). Metadata should be present if and only if the corresponding data is also present. The surfaces can be present whether or not the cortex data are present.

See the "Label Levels" section for the requirements of `xifti$meta$subcort$labels`.

Value

Logical. Is x a valid "xifti" object?

Label Levels

`xifti$meta$subcort$labels` is a factor with the following levels:

1. Cortex-L
2. Cortex-R
3. Accumbens-L
4. Accumbens-R
5. Amygdala-L
6. Amygdala-R
7. Brain Stem

8. Caudate-L
9. Caudate-R
10. Cerebellum-L
11. Cerebellum-R
12. Diencephalon-L
13. Diencephalon-R
14. Hippocampus-L
15. Hippocampus-R
16. Pallidum-L
17. Pallidum-R
18. Putamen-L
19. Putamen-R
20. Thalamus-L
21. Thalamus-R

These correspond to the same structures as given by `ft_read_cifti` in the `cifti-matlab` MATLAB toolbox.

`is.surf` *Validate a surface (vertices + faces)*

Description

Check if object is valid for `xifti$surf$cortex_left` or `xifti$surf$cortex_right`, where `xifti` is a `"xifti"` object.

Usage

```
is.surf(x)
```

Arguments

`x`	The putative surface.

Details

This is a helper function for `is.xifti`.

Requirements: the surface must be a list of three components: "vertices", "faces", and "hemisphere". The first two should each be a numeric matrix with three columns. The values in "vertices" represent spatial coordinates whereas the values in "faces" represent vertex indices defining the face. Thus, values in "faces" should be integers between 1 and the number of vertices. The last list entry, "hemisphere", should be "left", "right", or NULL indicating the brain hemisphere which the surface represents.

Value

Logical. Is x a valid surface?

`is.xifti` *Validate a "xifti" object.*

Description

Check if object is valid for a "xifti" object.

Usage

```
is.xifti(x, messages = TRUE)

is_xifti(x, messages = TRUE)
```

Arguments

x	The putative "xifti" object.
messages	If x is not a "xifti" object, print messages explaining the problem? Default is TRUE.

Details

Requirements: it is a list with the same structure as template_xifti. The size of each data entry must be compatible with its corresponding mask (medial wall for the cortex and volumetric mask for the subcortex). Metadata should be present if and only if the corresponding data is also present. The surfaces can be present whether or not the cortex data are present.

See the "Label Levels" section for the requirements of xifti$meta$subcort$labels.

Value

Logical. Is x a valid "xifti" object?

Label Levels

xifti$meta$subcort$labels is a factor with the following levels:

1. Cortex-L
2. Cortex-R
3. Accumbens-L
4. Accumbens-R
5. Amygdala-L
6. Amygdala-R
7. Brain Stem

8. Caudate-L
9. Caudate-R
10. Cerebellum-L
11. Cerebellum-R
12. Diencephalon-L
13. Diencephalon-R
14. Hippocampus-L
15. Hippocampus-R
16. Pallidum-L
17. Pallidum-R
18. Putamen-L
19. Putamen-R
20. Thalamus-L
21. Thalamus-R

These correspond to the same structures as given by `ft_read_cifti` in the `cifti-matlab` MATLAB toolbox.

`labels_Description` *labels*

Description

labels

Label Levels

`xifti$meta$subcort$labels` is a factor with the following levels:

1. Cortex-L
2. Cortex-R
3. Accumbens-L
4. Accumbens-R
5. Amygdala-L
6. Amygdala-R
7. Brain Stem
8. Caudate-L
9. Caudate-R
10. Cerebellum-L
11. Cerebellum-R

12. Diencephalon-L
13. Diencephalon-R
14. Hippocampus-L
15. Hippocampus-R
16. Pallidum-L
17. Pallidum-R
18. Putamen-L
19. Putamen-R
20. Thalamus-L
21. Thalamus-R

These correspond to the same structures as given by `ft_read_cifti` in the `cifti-matlab` MATLAB toolbox.

`make_color_pal` *Make a color palette.*

Description

Control the mapping of values to colors with `colors`, `color_mode`, and `zlim`.

Usage

```
make_color_pal(
  colors = NULL,
  color_mode = c("sequential", "qualitative", "diverging"),
  zlim = NULL,
  DATA_MIN = 0,
  DATA_MAX = 1
)
```

Arguments

`colors`	(Optional) "ROY_BIG_BL", the name of a ColorBrewer palette (see `RColorBrewer::brewer.pal.info` and colorbrewer2.org), or a character vector of colors. `NULL` (default) will use `"ROY_BIG_BL"` if `color_mode` is `"sequential"` or `"diverging"`, and `"Set2"` if `color_mode` is `"qualitative"`. See the description for more details.
`color_mode`	(Optional) `"sequential"`, `"qualitative"`, or `"diverging"`. Default: `"sequential"`. See the description for more details.
`zlim`	(Optional) Controls the mapping of values to each color in `colors`. If the length is longer than one, using `-Inf` will set the value to `DATA_MIN`, and `Inf` will set the value to `DATA_MAX`. See the description for more details.
`DATA_MIN`	(Optional) The minimum value of the data to make the palette for. Overrided by certain `zlim`.
`DATA_MAX`	(Optional) The maximum value of the data to make the palette for. Overrided by certain `zlim`.

Details

There are three argument types for `colors`: `"ROY_BIG_BL"`, the name of an `RColorBrewer` palette, or a character vector of color names.

If `colors=="ROY_BIG_BL"`, the "ROY_BIG_BL" pallete will be used. It is the same palette as the default used in the Connectome Workbench application (see github.com/Washington-University/workbench/blob/master/src/E The midpoint will be olored black. From the midpoint toward the upper bound, colors will proceed from black to red to yellow. From the midpoint toward the lower bound, colors will proceed from black to blue to purple to green to aqua. Note that these colors are not equally-spaced, and the bottom 0.5% of the color range has the same color. Here is how each color mode behaves if `colors=="ROY_BIG_BL"`:

`color_mode=="sequential"` Only the second half of the pallete will be used (black –> red –> yellow). If `identical(zlim,NULL)`, the colors will be mapped between `DATA_MIN` (black) to `DATA_MAX` (yellow). If `length(zlim)==2`, `zlim[1]` will be the lower bound (black) and `zlim[2]` will be the upper bound (yellow). If `zlim[1] > zlim[2]`, the first value will be used as the maximum and the second will be used as the minimum, and the color scale will be reversed with the highest value colored black and the lowest value colored yellow.

`color_mode=="qualitative"` The "ROY_BIG_BL" pallete is not recommended for qualitative data, so a warning will be issued. Colors will be based on the landmark colors in the "ROY_BIG_BL" pallete. If `identical(zlim,NULL)`, the colors will be mapped onto each integer between `DATA_MIN` and `DATA_MAX`, inclusive. Color interpolation will be used if the number of colors in the palette (17) is less than this range. If `length(zlim)==length(colors)`, each color will be mapped to each corresponding value.

`color_mode=="diverging"` If `identical(zlim,NULL)`, the colors will be mapped from `DATA_MIN` (aqua) to `DATA_MAX` (yellow). If `length(zlim)==1`, this value will be used as the midpoint (black) instead of the data midpoint. If `length(zlim)==2`, `zlim[1]` will be the lower bound (aqua) and `zlim[2]` will be the upper bound (yellow). If `length(zlim)==3`, these values will correspond to the lowest bound (aqua), midpoint (black), and upper bound (yellow) respectively. If the `zlim` are in descending order, the first value will be used as the maximum and the last will be used as the minimum, and the color scale will be reversed with the highest values colored aqua and the lowest values colored yellow.

If `colors` is the name of an RColorBrewer palette (see `RColorBrewer::brewer.pal.info`), the colors in that pallete will be used, and the following behavior applies. If `colors` is a character vector of color names (hex codes or standard R color names), the below behavior applies directly:

`color_mode=="sequential"` If `identical(zlim,NULL)`, the colors will be mapped with equal spacing from `DATA_MIN` to `DATA_MAX`. If `length(zlim)==2`, these values will be used as the upper and lower bounds instead. If `zlim[1] > zlim[2]`, the first value will be used as the maximum and the second will be used as the minimum, and the color scale will be reversed. If `length(zlim)==length(colors)`, each color will be mapped to each corresponding value.

`color_mode=="qualitative"` If `identical(zlim,NULL)`, the colors will be mapped onto each integer between `DATA_MIN` and `DATA_MAX`, inclusive. Color interpolation will be used if the number of colors in the palette is less than this range. If `length(zlim)==length(colors)`, each color will be mapped to each corresponding value.

`color_mode=="diverging"` If `identical(zlim,NULL)`, the colors will be mapped with equal spacing from `DATA_MIN` to `DATA_MAX`. Thus, the middle color will correspond to the midpoint

of the data. If `length(zlim)==1`, the middle color will correspond to this value instead. The preceeding colors will be equally-spaced between `DATA_MIN` and this value; the following colors will be equally-spaced between this value and`DATA_MAX`. If `length(zlim)==2`, `zlim[1]` will be the lower bound (first color) and `zlim[2]` will be the upper bound (last color). If `length(zlim)==3`, these values will correspond to the lowest bound, midpoint, and upper bound respectively. There must be an odd number of colors, since the diverging color mode requires a midpoint. If the `zlim` are in descending order, the first value will be used as the maximum and the last will be used as the minimum, and the color scale will be reversed. Finally, if `length(zlim)==length(colors)`, each color will be mapped to each corresponding value. Thus, the middle color will correspond to the middle color_value. The length of `colors` must be odd and >= 3.

Value

A data.frame with two columns: `"color"` (character: color hex codes) and `"value"` (numeric)

`make_surf` *Convert input to a* `"surf"` *object*

Description

Coerce a file path to a surface GIFTI, a `"gifti"` object, a list with entries "pointset" and "triangle", or a `"surf"` object to a `"surf"` object.

Usage

```
make_surf(surf, expected_hemisphere = NULL)

gifti_to_surf(surf, expected_hemisphere = NULL)
```

Arguments

`surf`	Either a file path to a surface GIFTI; a "gifti" object read by `readgii`; a list with entries "pointset" and "triangle"; or, a `"surf"` object.
`expected_hemisphere`	The expected hemisphere (`"left"` or `"right"`) of `surf`. If the hemisphere indicated in the GIFTI metadata is the opposite, an error is raised. If `NULL` (default), use the GIFTI hemisphere.

Value

The `"surf"` object: a list with components `"vertices"` (3D spatial locations), `"faces"` (defined by three vertices), and `"hemisphere"` (`"left"`, `"right"`, or `NULL` if unknown).

original_fnames_Param_resampled *original_fnames: for resampling*

Description

original_fnames: for resampling

Arguments

`original_fnames`
: The files to resample. This is a named list where each element's name is a file type label, and each element's value is a file name. Labels must be one of the following: "cortexL", "cortexR", "ROIcortexL", "ROIcortexR", "surfL", or "surfR". If `read_dir` is not `NULL`, then all these file names should be relative to `read_dir`.

parc_borders *Parcellation borders*

Description

Identify vertices which lie on the border of different parcels.

Usage

```
parc_borders(parc, surf = NULL, hemisphere = c("left", "right"))
```

Arguments

`parc`	Integer vector the same length as the number of vertices. Each entry indicates the parcel that vertex belongs to.
`surf`	The surface which the vertices belong to, or just the `"faces"` component ($Fx3$ matrix where each row indicates the vertices which comprise a face). If not provided, the (resampled) default `hemisphere` surface included with `ciftiTools` will be used.
`hemisphere`	Only used to choose which default surface to use if `is.null(surf)`. Should be `"left"` (default) or `"right"`.

Value

Logical vector the same length as `parc` indicating if the vertex lies on a border.

plot.surf *S3 method: plot surface*

Description

Visualize a single surface

Usage

```
## S3 method for class 'surf'
plot(x, ...)
```

Arguments

x	A "surf" object
...	Additional arguments to view_xifti_surface. But, the hemisphere argument behaves differently: it can be either left or right to indicate which hemisphere x represents. It is only used if the "hemisphere" metadata entry in x is NULL. If both the argument and the metadata entry are NULL, the surface will be treated as the left hemisphere.

plot.xifti *S3 method: use* view_xifti *to plot a* "xifti" *object*

Description

S3 method: use view_xifti to plot a "xifti" object

Usage

```
## S3 method for class 'xifti'
plot(x, ...)
```

Arguments

x	Object of class "xifti". See is.xifti and make_xifti.
...	Additional arguments to view_xifti, except what, which will be set to NULL.

read_cifti *Read a CIFTI file*

Description

Read a CIFTI file as a "xifti" object (see is.xifti).

Usage

```
read_cifti(
  cifti_fname = NULL,
  flat = FALSE,
  surfL_fname = NULL,
  surfR_fname = NULL,
  brainstructures = c("left", "right"),
  resamp_res = NULL,
  mwall_values = c(NA, NaN),
  wb_path = NULL,
  verbose = FALSE,
  ...
)

readCIfTI(
  cifti_fname,
  flat = FALSE,
  surfL_fname = NULL,
  surfR_fname = NULL,
  brainstructures = c("left", "right"),
  resamp_res = NULL,
  mwall_values = c(NA, NaN),
  wb_path = NULL,
  verbose = FALSE,
  ...
)

readcii(
  cifti_fname,
  flat = FALSE,
  surfL_fname = NULL,
  surfR_fname = NULL,
  brainstructures = c("left", "right"),
  resamp_res = NULL,
  mwall_values = c(NA, NaN),
  wb_path = NULL,
  verbose = FALSE,
  ...
)
```

```
read_xifti(
  cifti_fname,
  flat = FALSE,
  surfL_fname = NULL,
  surfR_fname = NULL,
  brainstructures = c("left", "right"),
  resamp_res = NULL,
  mwall_values = c(NA, NaN),
  wb_path = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

`cifti_fname`	File path of CIFTI-format data (ending in ".d*.nii").
`flat`	Should the result be flattened into a single matrix? If `FALSE` (default), the result will be a `"xifti"` object. If `TRUE`, the result will be a TxG matrix (T measurements, G greyordinates not including the medial wall if it's excluded from the ROI). All below arguments will be ignored because the brain structures cannot be identified. Surfaces will not be appended. Resampling is also not possible. `flat==TRUE` is the fastest way to read in just the CIFTI data. If `TRUE`, the greyordinates will be ordered by left cortex, right cortex, and then subcortex. Subcortical voxels will be ordered by alphabetical label. However, where each brainstructure (and subcortical structure) begins and ends cannot be determined. The medial wall locations and subcortical brain mask are also not included. The data matrix will be identical to that created by `-cifti-convert -to-gifti-ext`.
`surfL_fname`	(Optional) File path of GIFTI surface geometry file representing the left cortex.
`surfR_fname`	(Optional) File path of GIFTI surface geometry file representing the right cortex.
`brainstructures`	Character vector indicating which brain structure(s) to obtain: `"left"` (left cortical surface), `"right"` (right cortical surface) and/or `"subcortical"` (subcortical and cerebellar gray matter). Can also be `"all"` (obtain all three brain structures). Default: `c("left","right")` (cortical surface only). If a brain structure is indicated but does not exist, a warning will be raised and that brain structure will be skipped.
`resamp_res`	Resolution to resample the cortical data and surface to. Default: `NULL` (do not resample). If not `NULL`, the data will have to be read in with `-cifti-separate`, which is slower than `-cifti-convert -to-gifti-ext`.
`mwall_values`	If the medial wall locations are not indicated in the CIFTI, use these values to infer the medial wall mask. Default: `c(NA,NaN)`. If `NULL`, do not attempt to infer the medial wall.
`wb_path`	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with `ciftiTools.setOption("wb_path","path/to/workbench")`.

verbose	Should occasional updates be printed? Default: FALSE.
...	Additional arguments to read_cifti_convert or read_cifti_separate.

Details

First, metadata is obtained with info_cifti. Then, if no resampling is requested, the -cifti-convert -to-gifti-ext Workbench Command is used to "flatten" the data and save it as a metric GIFTI file, which is read in and separated by brainstructure according to the metadata (read_cifti_convert). Otherwise, if sampling is requested, then the CIFTI is separated into its GIFTI and NIFTI components, resampled, and then re-assembled (read_cifti_separate). The former is much faster for large CIFTI files, so the latter is only used when necessary for resampling.

If cifti_fname is not provided, then only the surfaces are read in.

Value

If !flat, a "xifti" object. Otherwise, a TxG matrix (T measurements, G greyordinates).

Connectome Workbench Requirement

This function uses a system wrapper for the 'wb_command' executable. The user must first download and install the Connectome Workbench, available from https://www.humanconnectome.org/software/get-connectome-workbench . The wb_path argument is the full file path to the Connectome Workbench folder. (The full file path to the 'wb_cmd' executable also works.)

Label Levels

xifti$meta$subcort$labels is a factor with the following levels:

1. Cortex-L
2. Cortex-R
3. Accumbens-L
4. Accumbens-R
5. Amygdala-L
6. Amygdala-R
7. Brain Stem
8. Caudate-L
9. Caudate-R
10. Cerebellum-L
11. Cerebellum-R
12. Diencephalon-L
13. Diencephalon-R
14. Hippocampus-L
15. Hippocampus-R
16. Pallidum-L

17. Pallidum-R
18. Putamen-L
19. Putamen-R
20. Thalamus-L
21. Thalamus-R

These correspond to the same structures as given by `ft_read_cifti` in the `cifti-matlab` MATLAB toolbox.

`read_dir_Param_separated` *read_dir: separated files*

Description

read_dir: separated files

Arguments

`read_dir`	Directory to append to the path of every file being read, e.g. `cortexL_original_fname`. If `NULL` (default), do not append any directory to the path. `read_dir` must already exist, or an error will be raised.

`resample_cifti` *Resample CIFTI Data*

Description

Performs spatial resampling of CIFTI data on the cortical surface by separating it into GIFTI and NIFTI files, resampling the GIFTIs, and then putting them together. (The subcortex is not resampled.)

Usage

```
resample_cifti(
  cifti_original_fname,
  cifti_target_fname,
  surfL_original_fname = NULL,
  surfR_original_fname = NULL,
  surfL_target_fname = NULL,
  surfR_target_fname = NULL,
  resamp_res,
  sep_keep = FALSE,
  sep_fnames = NULL,
```

```
  resamp_keep = FALSE,
  resamp_fnames = NULL,
  write_dir = NULL,
  mwall_values = c(NA, NaN),
  verbose = TRUE,
  wb_path = NULL
)

resampleCIfTI(
  cifti_original_fname,
  cifti_target_fname,
  surfL_original_fname = NULL,
  surfR_original_fname = NULL,
  surfL_target_fname = NULL,
  surfR_target_fname = NULL,
  resamp_res,
  sep_keep = FALSE,
  sep_fnames = NULL,
  resamp_keep = FALSE,
  resamp_fnames = NULL,
  write_dir = NULL,
  mwall_values = c(NA, NaN),
  verbose = TRUE,
  wb_path = NULL
)

resamplecii(
  cifti_original_fname,
  cifti_target_fname,
  surfL_original_fname = NULL,
  surfR_original_fname = NULL,
  surfL_target_fname = NULL,
  surfR_target_fname = NULL,
  resamp_res,
  sep_keep = FALSE,
  sep_fnames = NULL,
  resamp_keep = FALSE,
  resamp_fnames = NULL,
  write_dir = NULL,
  mwall_values = c(NA, NaN),
  verbose = TRUE,
  wb_path = NULL
)

resample_xifti(
  cifti_original_fname,
  cifti_target_fname,
  surfL_original_fname = NULL,
```

```
  surfR_original_fname = NULL,
  surfL_target_fname = NULL,
  surfR_target_fname = NULL,
  resamp_res,
  sep_keep = FALSE,
  sep_fnames = NULL,
  resamp_keep = FALSE,
  resamp_fnames = NULL,
  write_dir = NULL,
  mwall_values = c(NA, NaN),
  verbose = TRUE,
  wb_path = NULL
)
```

Arguments

`cifti_original_fname`
: A CIFTI file to resample.

`cifti_target_fname`
: The file name to save the resampled CIFTI.

`surfL_original_fname, surfR_original_fname`
: (Optional) File path of existing GIFTI surface geometry file representing the left/right cortex. One or both can be provided. These will be resampled too, and are convenient for visualizing the resampled data.

`surfL_target_fname, surfR_target_fname`
: (Optional) File path for the resampled GIFTI surface geometry file representing the left/right cortex. If NULL (default), will use default names: see `resample_cifti_default_fname`.

`resamp_res`
: Target resolution for resampling (number of cortical surface vertices per hemisphere).

`sep_keep`
: If separated files are created, should they be kept or deleted at the end of this function call? Default: `FALSE` (delete). Keeping the separated files may help speed up certain tasks, for example when repeatedly iterating over subjects–the CIFTI will only be separated once instead of at each iteration.

`sep_fnames`
: (Optional) Where to write the separated files (override their default file names). This is a named list where each entry's name is a file type label, and each entry's value is a file name indicating where to write the corresponding separated file. The recognized file type labels are: "cortexL", "cortexR", "ROIcortexL", "ROIcortexR", "subcortVol", and "subcortLabs".

  Entry values can be `NULL`, in which case a default file name will be used: see `cifti_component_suffix`. Default file names will also be used for files that need to be separated/written but without a corresponding entry in `sep_fnames`.

  Entries in `sep_fnames` will be ignored if they are not needed based on `[ROI_]brainstructures`. For example, if `brainstructures="left"`, then `sep_fnames$cortexR` will be ignored if specified.

  The `write_dir` argument can be used to place each separated file in the same directory.

resamp_keep — If resampled files are created, will they be kept or deleted at the end of this function call? Default: FALSE (delete). Keeping the resampled files may help speed up certain tasks, for example when repeatedly iterating over CIFTI files–resampling will only be done once instead of every new iteration.

resamp_fnames — Where to write the resampled files. This is a named list where each entry's name is a file type label, and each entry's value is a file name indicating where to write the corresponding resampled file. The recognized file type labels are: "cortexL", "cortexR", "ROIcortexL", "ROIcortexR", "validROIcortexL", and "validROIcortexR".

Entry values can be NULL, in which case a default file name will be used: see resample_cifti_default_fname. Default file names will also be used for files that need to be resampled/written but without a corresponding entry in resamp_fnames.

Entries in resamp_fnames will be ignored if they are not needed based on [ROI_]brainstructures. For example, if brainstructures="left", then resamp_fnames$cortexR will be ignored if specified.

The write_dir argument can be used to place each resampled file in the same directory.

write_dir — Where should any output files be written? NULL (default) will write them to the current working directory.

Files flagged for deletion will be written to a temporary directory, and thus are not affected by this argument. So if sep_keep is TRUE, the separated files will be written to write_dir, but if sep_keep is FALSE, they will be written to tempdir() and later deleted. resamp_keep works similarly.

For read_cifti_separate, the surface files (surfL or surfR) are deleted if resamp_keep is FALSE, so in this case they will be written to tempdir(). But for resample_cifti, the surface files are kept even if resamp_keep is FALSE, so they will always be written to write_dir.

Different subfolders for the separated, resampled, and final output files cannot be specified by write_dir. Instead, modify the individual file names in sep_fnames and resamp_fnames.

write_dir must already exist, or an error will occur.

mwall_values — If the medial wall locations are not indicated in the CIFTI, use these values to infer the medial wall mask. Default: c(NA,NaN). If NULL, do not attempt to infer the medial wall.

Correctly indicating the medial wall locations is important for resampling, because the medial wall mask is taken into account during resampling calculations.

verbose — Should occasional updates be printed? Default: TRUE.

wb_path — (Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with ciftiTools.setOption("wb_path","path/to/workbench").

Value

A named character vector of written files: "cifti" and potentially "surfL" (if surfL_original_fname was provided) and/or "surfR" (if surfR_original_fname was provided).

Connectome Workbench Requirement

This function uses a system wrapper for the 'wb_command' executable. The user must first download and install the Connectome Workbench, available from https://www.humanconnectome.org/software/get-connectome-workbench . The `wb_path` argument is the full file path to the Connectome Workbench folder. (The full file path to the 'wb_cmd' executable also works.)

`resample_cifti_from_template`

Resample a CIFTI from a template

Description

Resample a CIFTI from a template CIFTI using the `-cifti-resample` Connectome Workbench command.

Usage

```
resample_cifti_from_template(
  original_fname,
  template_fname,
  target_fname,
  wb_path = NULL
)
```

Arguments

`original_fname`	A CIFTI file to resample.
`template_fname`	A CIFTI file to use as the template.
`target_fname`	The file name to save the resampled CIFTI.
`wb_path`	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with `ciftiTools.setOption("wb_path","path/to/workbench")`.

Value

The `target_fname`, invisibly

Connectome Workbench Requirement

This function uses a system wrapper for the 'wb_command' executable. The user must first download and install the Connectome Workbench, available from https://www.humanconnectome.org/software/get-connectome-workbench . The `wb_path` argument is the full file path to the Connectome Workbench folder. (The full file path to the 'wb_cmd' executable also works.)

resample_gifti *Resample a GIFTI file (with its ROI)*

Description

Performs spatial resampling of GIFTI data on the cortical surface.

Usage

```
resample_gifti(
  original_fname,
  target_fname,
  hemisphere = c("left", "right"),
  file_type = NULL,
  original_res = NULL,
  resamp_res,
  ROIcortex_original_fname = NULL,
  ROIcortex_target_fname = NULL,
  read_dir = NULL,
  write_dir = NULL,
  wb_path = NULL
)

resampleGIfTI(
  original_fname,
  target_fname,
  hemisphere,
  file_type = NULL,
  original_res = NULL,
  resamp_res,
  ROIcortex_original_fname = NULL,
  ROIcortex_target_fname = NULL,
  read_dir = NULL,
  write_dir = NULL,
  wb_path = NULL
)

resamplegii(
  original_fname,
  target_fname,
  hemisphere,
  file_type = NULL,
  original_res = NULL,
  resamp_res,
  ROIcortex_original_fname = NULL,
  ROIcortex_target_fname = NULL,
  read_dir = NULL,
```

```
  write_dir = NULL,
  wb_path = NULL
)
```

Arguments

original_fname	The GIFTI file to resample.
target_fname	Where to save the resampled file.
hemisphere	"left" (default) or "right". An error will occur if the hemisphere indicated in the GIFTI metadata does not match.
file_type	"metric", "label", "surf", or NULL (default) to infer from original_fname.
original_res	The resolution of the original file. If NULL (default), infer from the file.
resamp_res	Target resolution for resampling (number of cortical surface vertices per hemisphere).
ROIcortex_original_fname	The name of the ROI file corresponding to original_fname. Leave as NULL (default) if this doesn't exist or shouldn't be resampled.
ROIcortex_target_fname	The name of the resampled ROI file. Only applicable if ROIcortex_original_fname is provided.
read_dir	Directory to append to the path of every file name in original_fname and ROIcortex_original_fname. If NULL (default), do not append any directory to the path.
write_dir	Directory to append to the path of every file name in target_fname and ROIcortex_target_fname. If NULL (default), do not append any directory to the path.
wb_path	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with ciftiTools.setOption("wb_path","path/to/workbench").

Value

The resampled GIFTI file name, invisibly

---

resample_surf *Resample a "surf" object*

Description

Resample a "surf" object by writing it to a GIFTI, using the Connectome Workbench to resample it, and then reading the new file.

Usage

```
resample_surf(
  surf,
  resamp_res,
  hemisphere = c("left", "right"),
  wb_path = NULL
)
```

Arguments

surf	A "surf" object
resamp_res	The desired resolution
hemisphere	"left" or "right". Only used if not indicated by surf$hemisphere. An error will be raised if it does not match the hemisphere indicated in the intermediate written GIFTI.
wb_path	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with ciftiTools.setOption("wb_path","path/to/workbench").

Value

The new surface

---

resamp_fnames_Param *resamp_fnames*

Description

resamp_fnames

Arguments

resamp_fnames	Where to write the resampled files. This is a named list where each entry's name is a file type label, and each entry's value is a file name indicating where to write the corresponding resampled file. The recognized file type labels are: "cortexL", "cortexR", "ROIcortexL", "ROIcortexR", "validROIcortexL", and "validROIcortexR". Entry values can be NULL, in which case a default file name will be used: see resample_cifti_default_fname. Default file names will also be used for files that need to be resampled/written but without a corresponding entry in resamp_fnames. Entries in resamp_fnames will be ignored if they are not needed based on [ROI_]brainstructures. For example, if brainstructures="left", then resamp_fnames$cortexR will be ignored if specified. The write_dir argument can be used to place each resampled file in the same directory.

`resamp_keep_Param` *resamp_keep*

Description

resamp_keep

Arguments

`resamp_keep`	If resampled files are created, will they be kept or deleted at the end of this function call? Default: `FALSE` (delete). Keeping the resampled files may help speed up certain tasks, for example when repeatedly iterating over CIFTI files–resampling will only be done once instead of every new iteration.

`resamp_res_Param_optional` *resamp_res: optional*

Description

resamp_res: optional

Arguments

`resamp_res`	(Optional) Target resolution for resampling (number of cortical surface vertices per hemisphere). If `NULL` (default) or `FALSE`, do not perform resampling.

`resamp_res_Param_required` *resamp_res: required*

Description

resamp_res: required

Arguments

`resamp_res`	Target resolution for resampling (number of cortical surface vertices per hemisphere).

`rgl_interactive_plots_Description`

Navigating and Embedding the Interactive Plots

Description

Navigating and Embedding the Interactive Plots

Navigating and Embedding the Interactive Plots

This function opens an interactive Open GL window rendered by `rgl`. If `save==TRUE` and `close_after_save==TRUE`, the window will be closed after the function call. Otherwise, it is kept open and the following information applies:

To navigate the plot, left click and drag the cursor to rotate. Use the scroll wheel or right click and drag to zoom. Press the scroll wheel and drag to change the field-of-view. Execute `snapshot` to save the current window as a .png file. Execute `rgl.close` to close the window. `rgl.viewpoint` can be used for programmatic navigation.

The Open GL window can be embedded as an htmlwidget in an R Markdown document using one of two methods. The first is executing `rglwidget` in the chunk where the plot is made. This first method should work within both the RStudio IDE and a knitted .html file. The second method is executing `setupKnitr` at the start of the document and then using the chunk option `webgl=TRUE` in the chunk where the plot is made. The second method is specifically for knitted .html files. Although the first method is the newest approach and is recommended by others, we used the second method in the `ciftiTools` vignette because the first is not compatible with htmlpreview. For both methods, the window still needs to be open to render the widget. Also for both methods, you will probably need to tweak the image dimensions e.g. `fig.width=8,fig.height=5` in the chunk options, because it uses the defaults from RMarkdown/Knitr instead of what makes sense based on the dimensions of the Open GL window.

For `view_xifti_surface`, if `length(idx) > 1`, this function will automaticaly return an htmlwidget using the first method, but with a `playwidget` wrapper to add a slider to control which column index is being displayed. All the meshes will be rendered on top of one another in the Open GL window, so only the widget will be useful for viewing the data interactively. Since it uses the first method, it will not be visible with htmlpreview. No additional call to `rglwidget` is necessary, but `rgl.close` must be called in a following chunk to close the Open GL window.

`rgl_static_plots_Description`

Embedding the Static Plots

Description

Embedding the Static Plots

Embedding the Static Plots

If `save==TRUE`, the plot(s) is written to a .png file. (For `view_xifti_surface`, if `length(idx) > 1`, each `idx` will be written to a separate image file.) You can use `include_graphics` to embed an image file in an R Markdown document. If `close_after_save==TRUE`, the return value of this function call is the name(s) of the image file(s) that were written, so it can be used directly to display the image.

There's an additional way to embed an image of this plot without writing a .png file: use `save==FALSE` and set the chunk options `rgl=TRUE,format="png"`. You will probably need to tweak the image dimensions e.g. `fig.width=8,fig.height=5` in the chunk options, because it uses the defaults from RMarkdown/Knitr instead of what makes sense based on the dimensions of the Open GL window.

`ROI_brainstructures_Param_LR` *ROI_brainstructures*

Description

ROI_brainstructures

Arguments

`ROI_brainstructures`
: Character vector indicating which ROIs should be obtained. `NULL` (default) to not get any ROIs. Otherwise, this should be a subset of the `brainstructures` argument.

  ROIs are typically the medial wall mask for the cortex and subcortical mask for the subcortex.

`ROY_BIG_BL` *"ROY_BIG_BL" color palette*

Description

"ROY_BIG_BL", the default palette from the Connectome Workbench.

Usage

```
ROY_BIG_BL(min = 0, max = 1, mid = NULL, pos_half = FALSE)
```

Arguments

min	The minimum value for the color mapping. As in the original palette, the last color (aqua) is actually placed at the bottom .5% between the minimum and maximum. Default: 0
max	The maximum value for the color mapping. If this value is lower than the minimum, the color mapping will be reversed. If this is equal to the minimum, a palette with only the color black will be returned. Default: 1.
mid	(Optional) The midpoint value for the color mapping. If NULL (default), the true midpoint is used.
pos_half	Use the positive half (black –> red –> yellow) only? Default: FALSE.

Details

Yields the landmark color hex codes and values for the "ROY_BIG_BL" palette. This is the same color palette as the default Connectome Workbench palette. Source: github.com/Washington-University/workbench/blob/master/src/Files/PaletteFile.cxx

Value

A data.frame with two columns: "color" (character: color hex codes) and "value" (numeric)

---

run_wb_cmd *Wrapper for Connectome Workbench Commands*

Description

Runs a Connectome Workbench command that has already been formatted.

Usage

```
run_wb_cmd(cmd, wb_path, intern = FALSE)
```

Arguments

cmd	The full command, beginning after the workbench path.
wb_path	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with ciftiTools.setOption("wb_path","path/to/workbench").
intern	Return printed output? If FALSE (default), return logical indicating success instead.

Value

If intern==FALSE, a logical indicating if the command finished successfully. If intern==TRUE, the printed output of the command.

separate_cifti *Separate a CIFTI file*

Description

Separate a CIFTI file into GIFTI files for the cortical data and NIFTI files for the subcortical data and labels. ROIs can also be written to indicate the medial wall mask (cortex) and volume mask (subcortex). This uses the Connectome Workbench command `-cifti-separate`.

Usage

```
separate_cifti(
  cifti_fname,
  brainstructures = c("left", "right"),
  cortexL_fname = NULL,
  cortexR_fname = NULL,
  subcortVol_fname = NULL,
  subcortLabs_fname = NULL,
  ROI_brainstructures = "all",
  ROIcortexL_fname = NULL,
  ROIcortexR_fname = NULL,
  ROIsubcortVol_fname = NULL,
  write_dir = NULL,
  wb_path = NULL
)

separateCIfTI(
  cifti_fname,
  brainstructures = c("left", "right"),
  cortexL_fname = NULL,
  cortexR_fname = NULL,
  subcortVol_fname = NULL,
  subcortLabs_fname = NULL,
  ROI_brainstructures = "all",
  ROIcortexL_fname = NULL,
  ROIcortexR_fname = NULL,
  ROIsubcortVol_fname = NULL,
  write_dir = NULL,
  wb_path = NULL
)

separatecii(
  cifti_fname,
  brainstructures = c("left", "right"),
  cortexL_fname = NULL,
  cortexR_fname = NULL,
  subcortVol_fname = NULL,
```

```
  subcortLabs_fname = NULL,
  ROI_brainstructures = "all",
  ROIcortexL_fname = NULL,
  ROIcortexR_fname = NULL,
  ROIsubcortVol_fname = NULL,
  write_dir = NULL,
  wb_path = NULL
)
```

Arguments

`cifti_fname` File path of CIFTI-format data (ending in ".d*.nii").

`brainstructures`
: Character vector indicating which brain structure(s) to obtain: `"left"` (left cortical surface), `"right"` (right cortical surface) and/or `"subcortical"` (subcortical and cerebellar gray matter). Can also be `"all"` (obtain all three brain structures). Default: `c("left","right")` (cortical surface only).

  If a brain structure is indicated but does not exist, a warning will be raised and that brain structure will be skipped.

`cortexL_fname, cortexR_fname`
: (Optional) GIFTI file names (*.[func/label].gii) to save the [left/right] cortex data to. If not provided, defaults to `"*[L/R].[func/label].gii"`, where * is the file name component of `cifti_fname`. Will be written in `write_dir`.

  dtseries and dscalar files should use "func", whereas dlabel files should use "label".

`subcortVol_fname, subcortLabs_fname`
: (Optional) NIFTI file names to save the subcortical [volume/labels] to. If not provided, defaults to `"*[/.labels].nii"`, where * is the file name component of `cifti_fname`. Will be written in `write_dir`.

`ROI_brainstructures`
: Which ROIs should be obtained? `"all"` (default) to obtain ROIs for each of the `brainstructures`. `NULL` to not obtain any ROIs. This should be a subset of `brainstructures`.

`ROIcortexL_fname, ROIcortexR_fname`
: (Optional) GIFTI file names (*.[func/label].gii) to save the [left/right] cortex ROI to. If not provided, defaults to `"*ROI_[L/R].[func/label].gii"`, where * is the file name component of `cifti_fname`. The cortical ROIs typically represent the medial wall mask, with values of 1 for in-ROI (non-medial wall) vertices and 0 for out-of-ROI (medial wall) vertices. Will be written in `write_dir`.

  dtseries and dscalar files should use "func", whereas dlabel files should use "label".

`ROIsubcortVol_fname`
: (Optional) NIFTI file names to save the subcortical ROI to. If not provided, defaults to `"*ROI.nii"`, where * is the file name component of `cifti_fname`. The subcortical ROI typically represents the volumetric mask for the entire subcortical structure, with values of 1 for in-ROI (in subcortex) voxels and 0 for out-of-ROI (not in subcortex) voxels. Will be written in `write_dir`.

write_dir — Where should the separated files be placed? NULL (default) will write them to the current working directory.

write_dir must already exist, or an error will occur.

wb_path — (Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with ciftiTools.setOption("wb_path","path/to/workbench").

Details

Time unit, start, and step (dtseries files) will not be written to the GIFTI/NIFTIs. Column names (dscalar files) will not be written to the GIFTIs, as well as label names and colors (dlabel files). (Haven't checked the NIFTIs yet.)

ROI/medial wall behavior: If there are 32k vertices in the left cortex with 3k representing the medial wall, then both cortexL_fname and ROIcortexL_fname will have 32k entries, 3k of which having a value of 0 indicating the medial wall. The non-medial wall entries will have the data values in cortexL_fname and a value of 1 in ROIcortexL_fname. Thus, exporting ROIcortexL_fname is vital if the data values include 0, because 0-valued non-medial wall vertices and medial wall vertices cannot be distinguished from one another within cortexL_fname alone.

Value

A named character vector with the file paths to the written NIFTI and GIFTI files

Connectome Workbench Requirement

This function uses a system wrapper for the 'wb_command' executable. The user must first download and install the Connectome Workbench, available from https://www.humanconnectome.org/software/get-connectome-workbench . The wb_path argument is the full file path to the Connectome Workbench folder. (The full file path to the 'wb_cmd' executable also works.)

---

sep_fnames_Param *sep_fnames*

Description

sep_fnames

Arguments

sep_fnames — (Optional) Where to write the separated files (override their default file names). This is a named list where each entry's name is a file type label, and each entry's value is a file name indicating where to write the corresponding separated file. The recognized file type labels are: "cortexL", "cortexR", "ROIcortexL", "ROIcortexR", "subcortVol", and "subcortLabs".

Entry values can be NULL, in which case a default file name will be used: see cifti_component_suffix. Default file names will also be used for files that need to be separated/written but without a corresponding entry in sep_fnames.

Entries in sep_fnames will be ignored if they are not needed based on [ROI_]brainstructures. For example, if brainstructures="left", then sep_fnames$cortexR will be ignored if specified.

The write_dir argument can be used to place each separated file in the same directory.

---

sep_keep_Param *sep_keep*

Description

sep_keep

Arguments

sep_keep	If separated files are created, should they be kept or deleted at the end of this function call? Default: FALSE (delete). Keeping the separated files may help speed up certain tasks, for example when repeatedly iterating over subjects–the CIFTI will only be separated once instead of at each iteration.

---

smooth_cifti *Smooth a CIFTI*

Description

Smooth CIFTI data. This uses the -cifti-smoothing command from Connectome Workbench.

Usage

```
smooth_cifti(
  x,
  cifti_target_fname = NULL,
  surface_sigma,
  volume_sigma,
  surfL_fname = NULL,
  surfR_fname = NULL,
  cerebellum_fname = NULL,
  subcortical_zeroes_as_NA = FALSE,
  cortical_zeroes_as_NA = FALSE,
  subcortical_merged = FALSE,
  wb_path = NULL
)

smoothCIfTI(
  x,
```

```
  cifti_target_fname,
  surface_sigma,
  volume_sigma,
  surfL_fname = NULL,
  surfR_fname = NULL,
  cerebellum_fname = NULL,
  subcortical_zeroes_as_NA = FALSE,
  cortical_zeroes_as_NA = FALSE,
  subcortical_merged = FALSE,
  wb_path = NULL
)

smoothcii(
  x,
  cifti_target_fname,
  surface_sigma,
  volume_sigma,
  surfL_fname = NULL,
  surfR_fname = NULL,
  cerebellum_fname = NULL,
  subcortical_zeroes_as_NA = FALSE,
  cortical_zeroes_as_NA = FALSE,
  subcortical_merged = FALSE,
  wb_path = NULL
)
```

Arguments

`x`	The `"xifti"` object or CIFTI file to smooth.
`cifti_target_fname`	The file name to save the smoothed CIFTI. If `NULL`, will be set to a file in a temporary directory.
`surface_sigma`	The sigma for the gaussian surface smoothing kernel, in mm
`volume_sigma`	The sigma for the gaussian volume smoothing kernel, in mm
`surfL_fname, surfR_fname`	(Required if the corresponding cortex is present) Surface GIFTI files for the left and right cortical surface
`cerebellum_fname`	(Optional) Surface GIFTI file for the cerebellar surface
`subcortical_zeroes_as_NA, cortical_zeroes_as_NA`	Should zero-values in the subcortical volume or cortex be treated as NA? Default: `FALSE`.
`subcortical_merged`	Smooth across subcortical structure boundaries? Default: `FALSE`.
`wb_path`	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with `ciftiTools.setOption("wb_path","path/to/workbench")`.

Details

If the CIFTI is a ".dlabel" file (intent 3007), then it will be converted to a ".dscalar" file because the values will no longer be integer indices. Unless the label values were ordinal, this is probably not desired so a warning will be printed.

The input can also be a `"xifti"` object.

Surfaces are required for each hemisphere in the CIFTI. If they are not provided, the inflated surfaces included in `"ciftiTools"` will be used.

Value

The `cifti_target_fname`, invisibly

Connectome Workbench Requirement

This function uses a system wrapper for the 'wb_command' executable. The user must first download and install the Connectome Workbench, available from https://www.humanconnectome.org/software/get-connectome-workbench . The `wb_path` argument is the full file path to the Connectome Workbench folder. (The full file path to the 'wb_cmd' executable also works.)

`substructure_table` *Substructure table*

Description

Table of labels for cortex hemispheres (left and right) and subcortical substructures. The names used by the CIFTI format and the names used by `ciftiTools` are given.

Usage

```
substructure_table()
```

Details

The names used by `ciftiTools` are based on those in `FT_READ_CIFTI` from the FieldTrip MATLAB toolbox.

Value

A data.frame with each substructure along the rows. The first column gives the CIFTI format name and the second column gives the `ciftiTools` name.

summary.surf *Summarise cifti objects*

Description

Summary method for class "surf"

Usage

```
## S3 method for class 'surf'
summary(object, ...)

## S3 method for class 'summary.surf'
print(x, ...)

## S3 method for class 'surf'
print(x, ...)
```

Arguments

object	Object of class "surf". See is.surf and make_surf.
...	further arguments passed to or from other methods.
x	bject of class "surf".

summary.xifti *Summarise cifti objects*

Description

Summary method for class "xifti"

Usage

```
## S3 method for class 'xifti'
summary(object, ...)

## S3 method for class 'summary.xifti'
print(x, ...)

## S3 method for class 'xifti'
print(x, ...)
```

Arguments

object	Object of class "xifti". See is.xifti and make_xifti.
...	further arguments passed to or from other methods.
x	Object of class "xifti". See is.xifti and make_xifti.

---

supported_intents *The NIFTI intents supported by* ciftiTools

---

Description

Table of CIFTI file types (NIFTI intents) supported By ciftiTools.

Usage

```
supported_intents()
```

Details

See https://www.nitrc.org/forum/attachment.php?attachid=334&group_id=454&forum_id=1955 for information about the different NIFTI intents.

Value

A data.frame with each supported file type along the rows, and column names "extension", "intent_code", "value", and "intent_name"

---

surface_plot_Params *surface plot*

Description

surface plot

Arguments

view	Which view to display: "lateral", "medial", or "both". If NULL (default), both views will be shown. Each view will be plotted in a separate panel row.
width, height	The dimensions of the RGL window, in pixels. If both are NULL (default), the dimensions will be set to 1000 (width) x 700 (height) for 1x1 and 2x2 subplots, 1500 x 525 for 2x1 subplots, and 500 x 700 for 1x2 subplots. These defaults are chosen to fit comfortably within a 1600 x 900 screen. Specyfing only one will set the other to maintain the same aspect ratio. Both can be specified to set the dimensions exactly.

zoom	Adjustment to size of brain meshes. Default: 3/5 (100% + 3/5*100% = 160% the original size).
bg	Background color. NULL will not color the background (white).
title	Optional title(s) for the plot(s). It will be printed at the top in a separate subplot with 1/4 the height of the brain cortex subplots. Default: NULL will use the time index (".dtseries") or name (.dscalar or .dlabel) of the data column being plotted. To use a custom title(s), use a length 1 character vector (same title for each plot) or length length(idx) character vector (different title for each plot). Set to an empty string "" to omit the title. If the title is non-empty but does not appear, cex.title may need to be lowered.
cex.title	Font size multiplier for the title. NULL (default) will use 2 for titles less than 20 characters long, and smaller sizes for increasingly longer titles.
text_color	Color for text in title and colorbar legend. Default: "black".
save	Save the plot to a .png file named by fname? Default: FALSE.
close_after_save	If save==TRUE, close the interactive Open GL window at the end of this function call? Default: TRUE.
fname	An identifier to use for naming the saved images ("[fname].png") or video frames ("[fname]_1.png", "[fname]_2.png", ...). Default: "xifti" for xifti_view_surface and "surf" for view_surf.
alpha	Transparency value for mesh coloring, between 0 and 1. Default: 1.0 (no transparency).
edge_color	Outline each edge in this color. Default: NULL (do not outline the edges).
vertex_size	Draw each vertex with this size. Default: 0 (do not draw the vertices).
vertex_color	Draw each vertex in this color. Default: "black". Vertices are only drawn if vertex_size > 0
mode	Deprecated: has no effect and will be removed. See save and close_after_save.

---

surfL_fname_Param *surfL_fname*

Description

surfL_fname

Arguments

surfL_fname	(Optional) File path of GIFTI surface geometry file representing the left cortex.

surfL_original_fname_Param *surfL_original_fname*

Description

surfL_original_fname

Arguments

surfL_original_fname
: (Optional) File path of GIFTI surface geometry file representing the left cortex.

surfL_Param_optional *surfL*

Description

surfL

Arguments

surfL
: (Optional) Left brain surface model. Can be a file path to a GIFTI surface geometry file (ends in "*.surf.gii"), a "gifti" object representing surface geometry, or a "surf" object.

surfL_target_fname_Param *surfL_target_fname*

Description

surfL_target_fname

Arguments

surfL_target_fname
: (Optional) File path to save the resampled GIFTI surface geometry file representing the left cortex at.

`surfR_fname_Param` *surfR_fname*

Description

surfR_fname

Arguments

`surfR_fname`	(Optional) File path of GIFTI surface geometry file representing the right cortex.

`surfR_original_fname_Param` *surfR_original_fname*

Description

surfR_original_fname

Arguments

`surfR_original_fname`	(Optional) File path of GIFTI surface geometry file representing the right cortex.

`surfR_Param_optional` *surfR*

Description

surfR

Arguments

`surfR`	(Optional) Right brain surface model. Can be a file path to a GIFTI surface geometry file (ends in "*.surf.gii"), a `"gifti"` object representing surface geometry, or a `"surf"` object.

surfR_target_fname_Param *surfR_target_fname*

Description

surfR_target_fname

Arguments

surfR_target_fname	(Optional) File path to save the resampled GIFTI surface geometry file representing the right cortex at.

sys_path *Format a path for* system.

Description

Right now, it escapes spaces and parentheses with "\\".

Usage

```
sys_path(R_path)
```

Arguments

R_path	The name of the file. It should be properly formatted: if it exists, file.exists(R_path) should be TRUE.

Value

The name of the file

unmask_cortex *Unmask cortex*

Description

Get cortex data with medial wall vertices

Usage

```
unmask_cortex(cortex, mwall, mwall_fill = NA)
```

Arguments

cortex	V vertices x T measurements matrix
mwall	Logical vector with T TRUE values.
mwall_fill	The fill value to use for medial wall vertices.

Value

The unmasked cortex data

unmask_vol *Undo a volumetric mask*

Description

Un-applies a mask to vectorized data to yield its volumetric representation. The mask and data should have compatible dimensions: the number of rows in dat should equal the number of locations within the mask.

Usage

```
unmask_vol(dat, mask, fill = NA)
```

Arguments

dat	Data matrix with locations along the rows and measurements along the columns. If only one set of measurements were made, this may be a vector.
mask	Volumetric binary mask. TRUE indicates voxels inside the mask.
fill	The value for locations outside the mask. Default: NA.

Value

The 3D or 4D unflattened volume array

use_color_pal *Use a color palette*

Description

Applies a palette to a data vector to yield a vector of colors.

Usage

```
use_color_pal(data_values, pal, color_NA = "white", indices = FALSE)
```

Arguments

data_values	The values to map to colors
pal	The palette to use to map values to colors
color_NA	The color to use for NA values. Default: "white".
indices	Return the numeric indices of colors in pal$value rather than the colors themselves. A value of 0 will be used for missing data. Default: FALSE.

Value

A character vector of color names (or integers if indices).

verbose_Param_FALSE *verbose: FALSE*

Description

verbose: FALSE

Arguments

verbose	Should occasional updates be printed? Default: FALSE.

verbose_Param_TRUE *verbose: TRUE*

Description

verbose: TRUE

Arguments

verbose	Should occasional updates be printed? Default: TRUE.

`view_surf` *View "surf" object(s)*

Description

Visualize one or two "surf" objects(s), or the "surf" component(s) in a "xifti" using an interactive Open GL window made with `rgl`. The `rgl` package is required.

Usage

```
view_surf(
  ...,
  view = c("both", "lateral", "medial"),
  width = NULL,
  height = NULL,
  zoom = NULL,
  bg = NULL,
  title = NULL,
  cex.title = NULL,
  text_color = "black",
  save = FALSE,
  close_after_save = TRUE,
  fname = "surf",
  alpha = 1,
  edge_color = NULL,
  vertex_color = NULL,
  vertex_size = 0,
  mode = NULL
)
```

Arguments

`...`	One of: A "surf" object; two "surf" objects; or, a "xifti" object. If a "surf" object has an empty "hemisphere" metadata entry, it will be set to the opposite side of the other's if known; otherwise, it will be set to the left side. If both are unknown, the first will be taken as the left and the second as the right.
`view`	Which view to display: "lateral", "medial", or "both". If NULL (default), both views will be shown. Each view will be plotted in a separate panel row.
`width`	The dimensions of the RGL window, in pixels. If both are NULL (default), the dimensions will be set to 1000 (width) x 700 (height) for 1x1 and 2x2 subplots, 1500 x 525 for 2x1 subplots, and 500 x 700 for 1x2 subplots. These defaults are chosen to fit comfortably within a 1600 x 900 screen. Specyfing only one will set the other to maintain the same aspect ratio. Both can be specified to set the dimensions exactly.
`height`	The dimensions of the RGL window, in pixels. If both are NULL (default), the dimensions will be set to 1000 (width) x 700 (height) for 1x1 and 2x2 subplots,

1500 x 525 for 2x1 subplots, and 500 x 700 for 1x2 subplots. These defaults are chosen to fit comfortably within a 1600 x 900 screen. Specyfing only one will set the other to maintain the same aspect ratio. Both can be specified to set the dimensions exactly.

`zoom` Adjustment to size of brain meshes. Default: `3/5` (100% + 3/5*100% = 160% the original size).

`bg` Background color. `NULL` will not color the background (white).

`title` Optional title(s) for the plot(s). It will be printed at the top in a separate subplot with 1/4 the height of the brain cortex subplots.

Default: `NULL` will use the time index (".dtseries") or name (.dscalar or .dlabel) of the data column being plotted.

To use a custom title(s), use a length 1 character vector (same title for each plot) or length `length(idx)` character vector (different title for each plot). Set to an empty string `""` to omit the title.

If the title is non-empty but does not appear, `cex.title` may need to be lowered.

`cex.title` Font size multiplier for the title. `NULL` (default) will use `2` for titles less than 20 characters long, and smaller sizes for increasingly longer titles.

`text_color` Color for text in title and colorbar legend. Default: "black".

`save` Save the plot to a .png file named by `fname`? Default: `FALSE`.

`close_after_save`

If `save==TRUE`, close the interactive Open GL window at the end of this function call? Default: `TRUE`.

`fname` An identifier to use for naming the saved images ("[fname].png") or video frames ("[fname]_1.png", "[fname]_2.png", ...). Default: `"xifti"` for `xifti_view_surface` and `"surf"` for `view_surf`.

`alpha` Transparency value for mesh coloring, between 0 and 1. Default: `1.0` (no transparency).

`edge_color` Outline each edge in this color. Default: `NULL` (do not outline the edges).

`vertex_color` Draw each vertex in this color. Default: `"black"`. Vertices are only drawn if `vertex_size > 0`

`vertex_size` Draw each vertex with this size. Default: `0` (do not draw the vertices).

`mode` Deprecated: has no effect and will be removed. See `save` and `close_after_save`.

Details

This function works as a wrapper to `view_xifti_surface`, but some arguments are not applicable (e.g. color scheme and legend). Also, instead of using the `hemisphere` argument, name the surface arguments `surfL` or `surfR` (see description for parameter `...`). Finally, the default value for `param` is `"surf"`, not `"xifti"`.

Navigating and Embedding the Interactive Plots

This function opens an interactive Open GL window rendered by `rgl`. If `save==TRUE` and `close_after_save==TRUE`, the window will be closed after the function call. Otherwise, it is kept open and the following information applies:

To navigate the plot, left click and drag the cursor to rotate. Use the scroll wheel or right click and drag to zoom. Press the scroll wheel and drag to change the field-of-view. Execute snapshot to save the current window as a .png file. Execute rgl.close to close the window. rgl.viewpoint can be used for programmatic navigation.

The Open GL window can be embedded as an htmlwidget in an R Markdown document using one of two methods. The first is executing rglwidget in the chunk where the plot is made. This first method should work within both the RStudio IDE and a knitted .html file. The second method is executing setupKnitr at the start of the document and then using the chunk option webgl=TRUE in the chunk where the plot is made. The second method is specifically for knitted .html files. Although the first method is the newest approach and is recommended by others, we used the second method in the ciftiTools vignette because the first is not compatible with htmlpreview. For both methods, the window still needs to be open to render the widget. Also for both methods, you will probably need to tweak the image dimensions e.g. fig.width=8,fig.height=5 in the chunk options, because it uses the defaults from RMarkdown/Knitr instead of what makes sense based on the dimensions of the Open GL window.

For view_xifti_surface, if length(idx) > 1, this function will automaticaly return an htmlwidget using the first method, but with a playwidget wrapper to add a slider to control which column index is being displayed. All the meshes will be rendered on top of one another in the Open GL window, so only the widget will be useful for viewing the data interactively. Since it uses the first method, it will not be visible with htmlpreview. No additional call to rglwidget is necessary, but rgl.close must be called in a following chunk to close the Open GL window.

Embedding the Static Plots

If save==TRUE, the plot(s) is written to a .png file. (For view_xifti_surface, if length(idx) > 1, each idx will be written to a separate image file.) You can use include_graphics to embed an image file in an R Markdown document. If close_after_save==TRUE, the return value of this function call is the name(s) of the image file(s) that were written, so it can be used directly to display the image.

There's an additional way to embed an image of this plot without writing a .png file: use save==FALSE and set the chunk options rgl=TRUE,format="png". You will probably need to tweak the image dimensions e.g. fig.width=8,fig.height=5 in the chunk options, because it uses the defaults from RMarkdown/Knitr instead of what makes sense based on the dimensions of the Open GL window.

---

view_xifti *View a "xifti" object*

Description

Switch for view_xifti_surface or view_xifti_volume

Usage

```
view_xifti(xifti, what = NULL, ...)

view_cifti(xifti, what = NULL, ...)
```

```
viewCIfTI(xifti, what = NULL, ...)

viewcii(xifti, what = NULL, ...)
```

Arguments

xifti	Object of class "xifti". See is.xifti and make_xifti.
what	Either "surf" or "volume". If NULL (default), view the surface if cortex data is present in the "xifti" object, and the subcortical volume otherwise.
...	Additional arguments to pass to either view function.

Value

The return value of view_xifti_surface or view_xifti_volume.

---

view_xifti_surface *View cortical surface*

Description

Visualize "xifti" cortical data using an interactive Open GL window made with rgl. The rgl and fields packages are required.

Usage

```
view_xifti_surface(
  xifti,
  idx = NULL,
  hemisphere = NULL,
  view = c("both", "lateral", "medial"),
  width = NULL,
  height = NULL,
  zoom = NULL,
  bg = NULL,
  title = NULL,
  cex.title = NULL,
  text_color = "black",
  save = FALSE,
  close_after_save = TRUE,
  fname = "xifti",
  colors = NULL,
  color_mode = NULL,
  zlim = NULL,
  surfL = NULL,
  surfR = NULL,
  qualitative_colorlegend = TRUE,
```

```
  colorlegend_ncol = NULL,
  colorbar_embedded = TRUE,
  colorbar_digits = NULL,
  alpha = 1,
  edge_color = NULL,
  vertex_color = NULL,
  vertex_size = 0,
  border_color = NULL,
  widget_idx_warn = NULL,
  render_rgl = TRUE,
  mode = NULL
)

view_cifti_surface(
  xifti,
  idx = NULL,
  hemisphere = NULL,
  view = c("both", "lateral", "medial"),
  width = NULL,
  height = NULL,
  zoom = NULL,
  bg = NULL,
  title = NULL,
  cex.title = NULL,
  text_color = "black",
  save = FALSE,
  close_after_save = TRUE,
  fname = "xifti",
  colors = NULL,
  color_mode = NULL,
  zlim = NULL,
  surfL = NULL,
  surfR = NULL,
  colorbar_embedded = TRUE,
  colorbar_digits = NULL,
  alpha = 1,
  edge_color = NULL,
  vertex_color = NULL,
  vertex_size = 0,
  widget_idx_warn = NULL,
  render_rgl = TRUE,
  mode = NULL
)

viewCIfTI_surface(
  xifti,
  idx = NULL,
  hemisphere = NULL,
```

```
  view = c("both", "lateral", "medial"),
  width = NULL,
  height = NULL,
  zoom = NULL,
  bg = NULL,
  title = NULL,
  cex.title = NULL,
  text_color = "black",
  save = FALSE,
  close_after_save = TRUE,
  fname = "xifti",
  colors = NULL,
  color_mode = NULL,
  zlim = NULL,
  surfL = NULL,
  surfR = NULL,
  colorbar_embedded = TRUE,
  colorbar_digits = NULL,
  alpha = 1,
  edge_color = NULL,
  vertex_color = NULL,
  vertex_size = 0,
  widget_idx_warn = NULL,
  render_rgl = TRUE,
  mode = NULL
)

viewcii_surface(
  xifti,
  idx = NULL,
  hemisphere = NULL,
  view = c("both", "lateral", "medial"),
  width = NULL,
  height = NULL,
  zoom = NULL,
  bg = NULL,
  title = NULL,
  cex.title = NULL,
  text_color = "black",
  save = FALSE,
  close_after_save = TRUE,
  fname = "xifti",
  colors = NULL,
  color_mode = NULL,
  zlim = NULL,
  surfL = NULL,
  surfR = NULL,
  colorbar_embedded = TRUE,
```

```
  colorbar_digits = NULL,
  alpha = 1,
  edge_color = NULL,
  vertex_color = NULL,
  vertex_size = 0,
  widget_idx_warn = NULL,
  render_rgl = TRUE,
  mode = NULL
)
```

Arguments

xifti	Object of class "xifti". See is.xifti and make_xifti.
idx	The time/column index of the data to display. If its length is greater than one and save==FALSE, use the widget instead of the OpenGL window to view the plot interactively, because the widget's slider will control what time/column is being displayed in the widget whereas all meshes will be rendered on top of one another in the Open GL window.
hemisphere	Which brain cortex to display: "both", "left", or "right". Each will be plotted in a separate panel column. If a brain cortex is requested but no surface is available, a default inflated surface will be used. This argument can also be NULL (default). In this case, the default inflated surface included with ciftiTools will be used for each cortex with data (i.e. if xifti$data$cortex_left and/or xifti$data$cortex_right exist). Surfaces without data will be colored white.
view	Which view to display: "lateral", "medial", or "both". If NULL (default), both views will be shown. Each view will be plotted in a separate panel row.
width	The dimensions of the RGL window, in pixels. If both are NULL (default), the dimensions will be set to 1000 (width) x 700 (height) for 1x1 and 2x2 subplots, 1500 x 525 for 2x1 subplots, and 500 x 700 for 1x2 subplots. These defaults are chosen to fit comfortably within a 1600 x 900 screen. Specyfing only one will set the other to maintain the same aspect ratio. Both can be specified to set the dimensions exactly.
height	The dimensions of the RGL window, in pixels. If both are NULL (default), the dimensions will be set to 1000 (width) x 700 (height) for 1x1 and 2x2 subplots, 1500 x 525 for 2x1 subplots, and 500 x 700 for 1x2 subplots. These defaults are chosen to fit comfortably within a 1600 x 900 screen. Specyfing only one will set the other to maintain the same aspect ratio. Both can be specified to set the dimensions exactly.
zoom	Adjustment to size of brain meshes. Default: 3/5 (100% + 3/5*100% = 160% the original size).
bg	Background color. NULL will not color the background (white).
title	Optional title(s) for the plot(s). It will be printed at the top in a separate subplot with 1/4 the height of the brain cortex subplots.

Default: NULL will use the time index (".dtseries") or name (.dscalar or .dlabel) of the data column being plotted.

To use a custom title(s), use a length 1 character vector (same title for each plot) or length length(idx) character vector (different title for each plot). Set to an empty string "" to omit the title.

If the title is non-empty but does not appear, cex.title may need to be lowered.

cex.title
: Font size multiplier for the title. NULL (default) will use 2 for titles less than 20 characters long, and smaller sizes for increasingly longer titles.

text_color
: Color for text in title and colorbar legend. Default: "black".

save
: Save the plot to a .png file named by fname? Default: FALSE.

close_after_save
: If save==TRUE, close the interactive Open GL window at the end of this function call? Default: TRUE.

fname
: An identifier to use for naming the saved images ("[fname].png") or video frames ("[fname]_1.png", "[fname]_2.png", ...). Default: "xifti" for xifti_view_surface and "surf" for view_surf.

colors
: (Optional) "ROY_BIG_BL", vector of colors to use, OR the name of a ColorBrewer palette (see RColorBrewer::brewer.pal.info and colorbrewer2.org). Defaults are "ROY_BIG_BL" (sequential), "Set2" (qualitative), and "ROY_BIG_BL" (diverging). An exception to these defaults is if the "xifti" object represents a .dlabel CIFTI (intent 3007), then the qualitative colors in the label table will be used. See the ciftiTools::make_color_pal() description for more details.

color_mode
: (Optional) "sequential", "qualitative", or "diverging". NULL will use the qualitative color mode if the "xifti" object represents a .dlabel CIFTI (intent 3007), and the sequential color mode otherwise. See the ciftiTools::make_color_pal() description for more details.

zlim
: (Optional) Controls the mapping of values to each color in colors. If the length is longer than one, using -Inf will set the value to DATA_MIN, and Inf will set the value to DATA_MAX. See the ciftiTools::make_color_pal() description for more details.

surfL, surfR
: (Optional if xifti$surf$cortex_left and xifti$surf$cortex_right are not empty) The brain surface model to use. Each can be a file path for a GIFTI, a file read by gifti::readgii, or a list with components "vertices" and "faces". If provided, they will override xifti$surf$cortex_left and xifti$surf$cortex_right if they exist. Otherwise, leave these arguments as NULL (default) to use xifti$surf$cortex_left and xifti$surf$cortex_right.

qualitative_colorlegend
: If color_mode=="qualitative", should the colorbar be replaced with a color legend? It will be printed separately from the RGL window. Default: TRUE.

colorlegend_ncol
: Number of columns in color legend. If NULL (default), use 10 entries per row. Only applies if the color legend is drawn (qualitative_colorlegend is TRUE and the qualitative color mode is used).

colorbar_embedded
: Should the colorbar be embedded in the plot? It will be positioned in the bottom-left corner, in a separate subplot with 1/4 the height of the brain cortex subplots. Default: TRUE. If FALSE, print it separately instead.

colorbar_digits
: The number of digits for the colorbar legend ticks. If NULL (default), let format decide.

alpha
: Transparency value for mesh coloring, between 0 and 1. Default: 1.0 (no transparency).

edge_color
: Outline each edge in this color. Default: NULL (do not outline the edges).

vertex_color
: Draw each vertex in this color. Default: "black". Vertices are only drawn if vertex_size > 0

vertex_size
: Draw each vertex with this size. Default: 0 (do not draw the vertices).

border_color
: Only applicable if color_mode is "qualitative". Border vertices will be identified (those that share a faces with at least one vertex of a different value) and drawn in this color instead, overriding the color indicated by their value. If NULL, do not draw borders.

widget_idx_warn
: If save==FALSE or close_after_save==FALSE, each mesh must be rendered in the same window. This is problematic if idx and the mesh size are large. So, this option can be used to print a warning when length(idx) exceeds widget_idx_warn and the meshes are being drawn in the same window. Use Inf to never print a warning. The default, NULL, will print a warning if idx times the number of vertices exceeds 200k (approximately three 32k meshes for both the left and right hemisphere.)

render_rgl
: Default: TRUE. If FALSE, do not render the Open GL window, widget, or image(s). Instead, return the rgl mesh objects and coloring information. This should be used by developers only.

mode
: Deprecated: has no effect and will be removed. See save and close_after_save.

Value

If save and close_after_save, the name(s) of the image file(s) that were written.

Otherwise, if the length of idx is equal to one, a list of the rgl object IDs which can be used to further manipulate the Open GL window; if the length of idx is greater than one, the htmlwidget with slider to interactively control which timepoint is being displayed.

Navigating and Embedding the Interactive Plots

This function opens an interactive Open GL window rendered by rgl. If save==TRUE and close_after_save==TRUE, the window will be closed after the function call. Otherwise, it is kept open and the following information applies:

To navigate the plot, left click and drag the cursor to rotate. Use the scroll wheel or right click and drag to zoom. Press the scroll wheel and drag to change the field-of-view. Execute snapshot to save the current window as a .png file. Execute rgl.close to close the window. rgl.viewpoint can be used for programmatic navigation.

The Open GL window can be embedded as an htmlwidget in an R Markdown document using one of two methods. The first is executing rglwidget in the chunk where the plot is made. This first method should work within both the RStudio IDE and a knitted .html file. The second method is executing setupKnitr at the start of the document and then using the chunk option webgl=TRUE

in the chunk where the plot is made. The second method is specifically for knitted .html files. Although the first method is the newest approach and is recommended by others, we used the second method in the `ciftiTools` vignette because the first is not compatible with htmlpreview. For both methods, the window still needs to be open to render the widget. Also for both methods, you will probably need to tweak the image dimensions e.g. `fig.width=8,fig.height=5` in the chunk options, because it uses the defaults from RMarkdown/Knitr instead of what makes sense based on the dimensions of the Open GL window.

For `view_xifti_surface`, if `length(idx) > 1`, this function will automaticaly return an htmlwidget using the first method, but with a `playwidget` wrapper to add a slider to control which column index is being displayed. All the meshes will be rendered on top of one another in the Open GL window, so only the widget will be useful for viewing the data interactively. Since it uses the first method, it will not be visible with htmlpreview. No additional call to `rglwidget` is necessary, but `rgl.close` must be called in a following chunk to close the Open GL window.

Embedding the Static Plots

If `save==TRUE`, the plot(s) is written to a .png file. (For `view_xifti_surface`, if `length(idx) > 1`, each `idx` will be written to a separate image file.) You can use `include_graphics` to embed an image file in an R Markdown document. If `close_after_save==TRUE`, the return value of this function call is the name(s) of the image file(s) that were written, so it can be used directly to display the image.

There's an additional way to embed an image of this plot without writing a .png file: use `save==FALSE` and set the chunk options `rgl=TRUE,format="png"`. You will probably need to tweak the image dimensions e.g. `fig.width=8,fig.height=5` in the chunk options, because it uses the defaults from RMarkdown/Knitr instead of what makes sense based on the dimensions of the Open GL window.

`view_xifti_volume` *View subcortex*

Description

Visualize subcortex of a `"xifti"` object

Usage

```
view_xifti_volume(
  xifti,
  structural_img = "MNI",
  idx = 1,
  plane = c("axial", "sagittal", "coronal"),
  num.slices = 9,
  use_papaya = FALSE,
  z_min = NULL,
  z_max = NULL,
  verbose = TRUE,
  ...
)
```

```
view_cifti_volume(
  xifti,
  structural_img = "MNI",
  idx = 1,
  plane = c("axial", "sagittal", "coronal"),
  num.slices = 9,
  use_papaya = FALSE,
  z_min = NULL,
  z_max = NULL,
  verbose = TRUE,
  ...
)

viewCIfTI_volume(
  xifti,
  structural_img = "MNI",
  idx = 1,
  plane = c("axial", "sagittal", "coronal"),
  num.slices = 9,
  use_papaya = FALSE,
  z_min = NULL,
  z_max = NULL,
  verbose = TRUE,
  ...
)

viewcii_volume(
  xifti,
  structural_img = "MNI",
  idx = 1,
  plane = c("axial", "sagittal", "coronal"),
  num.slices = 9,
  use_papaya = FALSE,
  z_min = NULL,
  z_max = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

`xifti`	Object of class "xifti". See `is.xifti` and `make_xifti`.
`structural_img`	The structural MRI image on which to overlay the subcortical values. Can be a file name, `"MNI"` (default) to use the MNI T1-weighted template, or `NULL` to use a blank image.
`idx`	The time/column index of the `"xifti"` data to plot.

plane	If use_papaya=FALSE, the plane to display. Default: "axial". Other options are "sagittal" and "coronal".
num.slices	If use_papaya=FALSE, the number of slices to display. Default: 9.
use_papaya	use_papaya=TRUE will use papayar to allows for interactive visualization.
z_min	Floor value.
z_max	Ceiling value.
verbose	Should occasional updates be printed? Default: TRUE.
...	Additional arguments to pass to papayar::papaya or oro.nifti::overlay

---

wb_path_Param *wb_path*

Description

wb_path

Arguments

wb_path	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with ciftiTools.setOption("wb_path","path/to/workbench").

---

write_cifti *Write a CIFTI file from a "xifti" object.*

Description

Write out a "xifti" object as a CIFTI file and (optionally) GIFTI surface files.

Usage

```
write_cifti(
  xifti,
  cifti_fname,
  surfL_fname = NULL,
  surfR_fname = NULL,
  verbose = TRUE,
  wb_path = NULL
)

writeCIfTI(
  xifti,
  cifti_fname,
  surfL_fname = NULL,
  surfR_fname = NULL,
```

```
  verbose = TRUE,
  wb_path = NULL
)

writecii(
  xifti,
  cifti_fname,
  surfL_fname = NULL,
  surfR_fname = NULL,
  verbose = TRUE,
  wb_path = NULL
)

write_xifti(
  xifti,
  cifti_fname,
  surfL_fname = NULL,
  surfR_fname = NULL,
  verbose = TRUE,
  wb_path = NULL
)
```

Arguments

`xifti`	Object of class "xifti". See `is.xifti` and `make_xifti`.
`cifti_fname`	File path of CIFTI-format data (ending in ".d*.nii").
`surfL_fname, surfR_fname`	If the [left/right] surface is present, it will be a written to a GIFTI file at this file path. If `NULL` (default), do not write out the surface.
`verbose`	Should occasional updates be printed? Default: `TRUE`.
`wb_path`	(Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with `ciftiTools.setOption("wb_path","path/to/workbench")`.

Value

Named character vector of the written files

Connectome Workbench Requirement

This function uses a system wrapper for the 'wb_command' executable. The user must first download and install the Connectome Workbench, available from https://www.humanconnectome.org/software/get-connectome-workbench . The `wb_path` argument is the full file path to the Connectome Workbench folder. (The full file path to the 'wb_cmd' executable also works.)

`write_dir_Param_generic`
write_dir: generic

Description

write_dir: generic

Arguments

`write_dir`	Where should the separated files be placed? `NULL` (default) will write them to the current working directory. `write_dir` must already exist, or an error will occur.

`write_dir_Param_intermediate`
write_dir: intermediate separated/resampled files

Description

write_dir: intermediate separated/resampled files

Arguments

`write_dir`	Where should any output files be written? `NULL` (default) will write them to the current working directory. Files flagged for deletion will be written to a temporary directory, and thus are not affected by this argument. So if `sep_keep` is `TRUE`, the separated files will be written to `write_dir`, but if `sep_keep` is `FALSE`, they will be written to `tempdir()` and later deleted. `resamp_keep` works similarly. For `read_cifti_separate`, the surface files (`surfL` or `surfR`) are deleted if `resamp_keep` is `FALSE`, so in this case they will be written to `tempdir()`. But for `resample_cifti`, the surface files are kept even if `resamp_keep` is `FALSE`, so they will always be written to `write_dir`. Different subfolders for the separated, resampled, and final output files cannot be specified by `write_dir`. Instead, modify the individual file names in `sep_fnames` and `resamp_fnames`. `write_dir` must already exist, or an error will occur.

`write_metric_gifti` *Write CIFTI cortex data to GIFTI*

Description

Write the data for the left or right cortex to a metric GIFTI file.

Usage

```
write_metric_gifti(
  x,
  gifti_fname,
  hemisphere = c("left", "right"),
  intent = NULL,
  data_type = NULL,
  encoding = NULL,
  endian = c("LittleEndian", "BigEndian"),
  col_names = NULL,
  label_table = NULL
)
```

Arguments

`x`	A VxT data matrix (V vertices, T measurements). This can also be an object from `gifti::readgii`, or a length T list of length V vectors.
`gifti_fname`	Where to write the GIFTI file.
`hemisphere`	`"left"` (default) or `"right"`. Ignored if `data` is already a `"gifti"` object.
`intent`	"NIFTI_INTENT_*". `NULL` (default) will use metadata if `data` is a `"gifti"` object, or "NONE" if it cannot be inferred. If not `NULL` and `data` is a `"gifti"` object, it will overwrite the existing intent. See https://nifti.nimh.nih.gov/nifti-1/documentation/nifti1fields/nifti1fields_pages/group__NIFTI1__INTENT__CODES.html/document_vi .
`data_type`	the type of `data`: "NIFTI_TYPE_*" where * is "INT32" or "FLOAT32". If `NULL` (default), the data type will be inferred. If not `NULL` and `data` is a `"gifti"` object, it will overwrite the existing data type.
`encoding`	One of "ASCII", "Base64Binary", or "GZipBase64Binary". If `NULL` (default), will use the metadata if `data` is a GIFTI object, or "ASCII" if the `data_type` is "NIFTI_TYPE_INT32" and "GZipBase64Binary" if the `data_type` is "NIFTI_TYPE_FLOAT32". If not `NULL` and `data` is a `"gifti"` object, it will overwrite the existing data type.
`endian`	"LittleEndian" (default) or "BigEndian". If `data` is a `"gifti"` object, it will overwrite the existing endian.
`col_names`	The names of each data column in `gii` (or entries in `gii$data`).

label_table — A data.frame with labels along rows. The row names should be the label names. The column names should be among: "Key", "Red", "Green", "Blue", and "Alpha". The "Key" column is required whereas the others are optional (but very often included). Values in the "Key" column should be non-negative integers, typically beginning with 0. The other columns should be floating-point numbers between 0 and 1.

Although CIFTI files support a different label table for each data column, GIFTI files only support a single label table. So this label table should be applicable to each data column.

Value

Whether the GIFTI was successfully written

`write_subcort_nifti` *Write subcortical data to NIFTI files*

Description

Write subcortical data to NIFTI files representing the data values, subcortical structure labels, and volumetric mask. The input formats of `subcortVol`, `subcortLabs`, and `subcortMask` correspond to the data structures of `xifti$data$subcort`, `xifti$meta$subcort$labels` , and `xifti$meta$subcort$mask` respectively. `subcortVol` and `subcortLabs` should be vectorized, so if they are volumes consider using `RNifti::writeNIfTI`.

Usage

```
write_subcort_nifti(
  subcortVol,
  subcortLabs,
  subcortMask,
  trans_mat = NULL,
  subcortVol_fname,
  subcortLabs_fname,
  ROIsubcortVol_fname = NULL,
  fill = 0,
  wb_path = NULL
)
```

Arguments

subcortVol — A vectorized data matrix: V voxels by T measurements

subcortLabs — Numeric (0 and 3-21) or factor vector corresponding to subcortical structure labels. See `substructure_table`.

subcortMask — Logical volumetric mask. Values of 0 represent out-of-mask voxels (not subcortical), and values of 1 represent in-mask voxels (subcortical),

trans_mat — The TransformationMatrixIJKtoXYZ, or equivalently the desired sform matrix (srow_x, srow_y and srow_z) to write. If NULL, do not write it (all zeroes).

subcortVol_fname, subcortLabs_fname, ROIsubcortVol_fname — File path to a NIFTI to save the corresponding data. ROIsubcortVol_fname is optional but the rest is required.

fill — Values to use for out-of-mask voxels. Default: 0.

wb_path — (Optional) Path to Connectome Workbench folder or executable. If not provided, should be set with ciftiTools.setOption("wb_path","path/to/workbench").

Details

All file path arguments are required except ROIsubcortVol_fname. If not provided, the volumetric mask will not be written. (It's redundant with the 0 values in subcortLabs_fname because valid labels have positive indexes.)

Value

Named character vector with the "subcortVol", "subcortLabs", and "ROIsubcortVol" file names (if written)

---

write_surf_gifti *Write CIFTI surface data to GIFTI*

Description

Write the data for the left or right surface to a surface GIFTI file.

Usage

```
write_surf_gifti(
  x,
  gifti_fname,
  hemisphere = c("left", "right"),
  encoding = NULL,
  endian = c("LittleEndian", "BigEndian")
)
```

Arguments

x — A "surf" object, an object from gifti::readgii, or a list with elements "pointset" and "triangle".

gifti_fname — Where to write the GIFTI file.

hemisphere — "left" (default) or "right". Ignored if data is already a "gifti" object, or if it is a "surf" object with the hemisphere metadata already specified.

encoding	A length-2 vector with elements chosen among "ASCII", "Base64Binary", and "GZipBase64Binary". If NULL (default), will use the metadata if data is a "gifti" object, or "GZipBase64Binary" for the "pointset" and "ASCII" for the "triangles" if data is not already a GIFTI.
endian	"LittleEndian" (default) or "BigEndian".

Value

Whether the GIFTI was successfully written

`xifti_Param` *xifti*

Description

xifti

Arguments

xifti	Object of class "xifti". See `is.xifti` and `make_xifti`.

`x_Param_xifti` *x: xifti*

Description

x: xifti

Arguments

x	Object of class "xifti". See `is.xifti` and `make_xifti`.

Index