

Package: wbcmd (via r-universe)

May 31, 2026

Title Wrapper Functions for 'Connectome Workbench' Commands

Version 0.1.0

Description Provides R wrapper functions for the 'Connectome Workbench' command-line tool ('wb_command'). Includes a central dispatcher for executing workbench commands, environment detection, and convenience wrappers for common operations on Connectivity Informatics Technology Initiative (CIFTI), Geometry Informatics Technology Initiative (GIFTI), and surface files.

License MIT + file LICENSE

URL <https://lcbc-uio.github.io/wbcmd/>,
<https://github.com/lcbc-uio/wbcmd>

BugReports <https://github.com/lcbc-uio/wbcmd/issues>

Imports cli, lifecycle, rlang, roxygen2, tools, utils

Suggests covr, knitr, pkgdown, rmarkdown, spelling, testthat (>= 3.0.0), withr

SystemRequirements Connectome Workbench
(<https://www.humanconnectome.org/software/connectome-workbench>)

Config/testthat/edition 3

Config/testthat/parallel true

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

VignetteBuilder knitr

Config/pak/sysreqs cmake make libuv1-dev libxml2-dev

Repository <https://ggsegverse.r-universe.dev>

Date/Publication 2026-02-23 21:54:53 UTC

RemoteUrl <https://github.com/lcbc-uio/wbcmd>

RemoteRef HEAD

RemoteSha 855466d27f77f2d1f95128930bfd21ebee1231ff

Contents

check_wb_result	2
cifti_average	3
cifti_correlation	4
cifti_dilate	5
cifti_find_clusters	7
cifti_gradient	8
cifti_math	9
cifti_parcellate	11
cifti_reduce	12
cifti_stats	13
cifti_weighted_stats	14
file_information	15
get_wb_path	16
get_wb_setting	17
get_wb_verbosity	17
have_wb	18
metric_math	18
metric_stats	19
set_wb_path	20
surface_curvature	21
surface_distortion	22
surface_generate_inflated	23
surface_geodesic_distance	24
surface_information	25
surface_vertex_areas	26
volume_math	27
volume_to_surface_mapping	28
wb_cmd	29
wb_help	30
wb_help_rd	31
wb_sitrep	31
wb_version	32

Index	33
--------------	-----------

check_wb_result	<i>Check the result of a workbench command</i>
-----------------	--

Description

Validates the outcome of a `wb_command` execution by inspecting the exit status and whether the expected output file was produced.

Usage

```
check_wb_result(  
  res,  
  fe_before = NA,  
  fe_after = NA,  
  outfile = NULL,  
  cmd_name = NULL  
)
```

Arguments

res	The return value from <code>system()</code> .
fe_before	Logical; did the output file exist before execution?
fe_after	Logical; does the output file exist after execution?
outfile	Path to the expected output file.
cmd_name	The command name, for diagnostic messages.

Value

res, invisibly.

Examples

```
check_wb_result(0L, fe_before = FALSE, fe_after = TRUE,  
  outfile = "out.nii", cmd_name = "test")
```

cifti_average	<i>Average multiple CIFTI files</i>
---------------	-------------------------------------

Description

Wraps `wb_command -cifti-average` to compute the element-wise average across multiple CIFTI files.

Usage

```
cifti_average(  
  cifti_out,  
  cifti_in,  
  weights = NULL,  
  verbose = get_wb_verbosity()  
)
```

Arguments

cifti_out	Path for the output averaged CIFTI file.
cifti_in	Character vector of input CIFTI file paths.
weights	Optional numeric vector of weights (same length as cifti_in).
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-average")` in a session with `wb_command` available.

Examples

```
## Not run:
cifti_average(
  "group_mean.dscalar.nii",
  c("subj01.dscalar.nii", "subj02.dscalar.nii", "subj03.dscalar.nii")
)

## End(Not run)
```

cifti_correlation	<i>Compute correlation of CIFTI rows</i>
-------------------	--

Description

Wraps `wb_command -cifti-correlation` to generate a row-wise correlation matrix from a CIFTI file.

Usage

```
cifti_correlation(
  cifti_in,
  cifti_out,
  roi_override = NULL,
  weights = NULL,
  fisher_z = FALSE,
  no_demean = FALSE,
  covariance = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

cifti_in	Path to the input CIFTI file.
cifti_out	Path for the output correlation CIFTI file.
roi_override	Path to a CIFTI ROI file to restrict computation.
weights	Path to a text file of column weights.
fisher_z	Logical; apply Fisher z-transform to correlations?
no_demean	Logical; skip demeaning before correlation?
covariance	Logical; compute covariance instead of correlation?
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-correlation")` in a session with `wb_command` available.

Examples

```
## Not run:  
cifti_correlation(  
  "data.dtseries.nii",  
  "corr.dconn.nii",  
  fisher_z = TRUE  
)  
  
## End(Not run)
```

`cifti_dilate`*Dilate a CIFTI file*

Description

Wraps `wb_command -cifti-dilate` to fill in zeros or missing data with values from nearby brain-ordinates.

Usage

```
cifti_dilate(
  cifti_in,
  direction = c("COLUMN", "ROW"),
  surface_distance,
  volume_distance,
  cifti_out,
  left_surface = NULL,
  right_surface = NULL,
  nearest = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

cifti_in	Path to the input CIFTI file.
direction	Direction to dilate. One of "COLUMN" or "ROW".
surface_distance	Surface dilation distance in mm.
volume_distance	Volume dilation distance in mm.
cifti_out	Path for the output CIFTI file.
left_surface	Path to the left hemisphere surface.
right_surface	Path to the right hemisphere surface.
nearest	Logical; use nearest good value instead of weighted average?
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-dilate")` in a session with `wb_command` available.

Examples

```
## Not run:
cifti_dilate(
  "data.dscalar.nii",
  direction = "COLUMN",
  surface_distance = 10,
  volume_distance = 10,
  cifti_out = "dilated.dscalar.nii",
  left_surface = "left.midthickness.surf.gii",
  right_surface = "right.midthickness.surf.gii"
)
```

```
## End(Not run)
```

```
cifti_find_clusters Find clusters in a CIFTI file
```

Description

Wraps `wb_command -cifti-find-clusters` to threshold a CIFTI file and filter clusters by surface area and volume size.

Usage

```
cifti_find_clusters(
  cifti_in,
  surface_value_threshold,
  surface_minimum_area,
  volume_value_threshold,
  volume_minimum_size,
  direction = c("COLUMN", "ROW"),
  cifti_out,
  left_surface = NULL,
  right_surface = NULL,
  less_than = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

<code>cifti_in</code>	Path to the input CIFTI file.
<code>surface_value_threshold</code>	Minimum value for surface vertices.
<code>surface_minimum_area</code>	Minimum cluster surface area (mm ²).
<code>volume_value_threshold</code>	Minimum value for volume voxels.
<code>volume_minimum_size</code>	Minimum cluster volume size (mm ³).
<code>direction</code>	Direction to use. One of "ROW" or "COLUMN".
<code>cifti_out</code>	Path for the output CIFTI file.
<code>left_surface</code>	Path to the left hemisphere surface.
<code>right_surface</code>	Path to the right hemisphere surface.
<code>less_than</code>	Logical; find clusters below threshold instead of above?
<code>verbose</code>	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-find-clusters")` in a session with `wb_command` available.

Examples

```
## Not run:
cifti_find_clusters(
  "zstats.dscalar.nii",
  surface_value_threshold = 2.3,
  surface_minimum_area = 100,
  volume_value_threshold = 2.3,
  volume_minimum_size = 200,
  direction = "COLUMN",
  cifti_out = "clusters.dscalar.nii",
  left_surface = "left.midthickness.surf.gii",
  right_surface = "right.midthickness.surf.gii"
)

## End(Not run)
```

cifti_gradient

Compute gradient of a CIFTI file

Description

Wraps `wb_command -cifti-gradient` to compute the spatial gradient of each column in a CIFTI file.

Usage

```
cifti_gradient(
  cifti_in,
  direction = c("COLUMN", "ROW"),
  cifti_out,
  left_surface = NULL,
  right_surface = NULL,
  surface_presmooth = NULL,
  volume_presmooth = NULL,
  average_output = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

cifti_in	Path to the input CIFTI file.
direction	Direction to take gradient. One of "COLUMN" or "ROW".
cifti_out	Path for the output gradient CIFTI file.
left_surface	Path to the left hemisphere surface.
right_surface	Path to the right hemisphere surface.
surface_presmooth	Surface pre-smoothing sigma in mm.
volume_presmooth	Volume pre-smoothing sigma in mm.
average_output	Logical; average the gradient across all maps?
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-gradient")` in a session with `wb_command` available.

Examples

```
## Not run:
cifti_gradient(
  "data.dscalar.nii",
  direction = "COLUMN",
  cifti_out = "gradient.dscalar.nii",
  left_surface = "left.midthickness.surf.gii",
  right_surface = "right.midthickness.surf.gii"
)

## End(Not run)
```

cifti_math

Evaluate expression on CIFTI files

Description

Wraps `wb_command -cifti-math` to perform element-wise mathematical operations on CIFTI files using an expression language.

Usage

```
cifti_math(  
  expression,  
  cifti_out,  
  var,  
  fixnan = FALSE,  
  override_mapping_check = FALSE,  
  verbose = get_wb_verbosity()  
)
```

Arguments

expression	Math expression string (e.g. "(x - mean) / stdev").
cifti_out	Path for the output CIFTI file.
var	Named list of input CIFTI file paths. Names correspond to variable names used in expression.
fixnan	Logical; replace NaN results with 0?
override_mapping_check	Logical; skip map count check?
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-math")` in a session with `wb_command` available.

Examples

```
## Not run:  
cifti_math(  
  "(x - y) / y",  
  cifti_out = "pct_change.dscalar.nii",  
  var = list(x = "post.dscalar.nii", y = "pre.dscalar.nii")  
)  
  
## End(Not run)
```

cifti_parcellate	<i>Parcellate a CIFTI file</i>
------------------	--------------------------------

Description

Wraps `wb_command -cifti-parcellate` to create parcellated data from a dense CIFTI file using a label parcellation.

Usage

```
cifti_parcellate(
  cifti_in,
  cifti_label,
  direction = c("COLUMN", "ROW"),
  cifti_out,
  spatial_weights = NULL,
  method = NULL,
  only_numeric = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

<code>cifti_in</code>	Path to the input CIFTI file.
<code>cifti_label</code>	Path to the parcellation label file (<code>.dlabel.nii</code>).
<code>direction</code>	Direction to parcellate along. One of "COLUMN" or "ROW".
<code>cifti_out</code>	Path for the output parcellated CIFTI file.
<code>spatial_weights</code>	Optional spatial weights specification. A list with one element: either <code>left_area_surf</code> and <code>right_area_surf</code> paths, or <code>cifti_weights</code> path.
<code>method</code>	Parcellation method. One of "MEAN" (default), "MAX", "MIN", "INDEXMAX", "INDEXMIN", "MEDIAN", "MODE", "COUNT_NONZERO", "SUM", "STDEV", "SAMPSTDEV", "PERCENTILE".
<code>only_numeric</code>	Logical; exclude non-numeric values?
<code>verbose</code>	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-parcellate")` in a session with `wb_command` available.

Examples

```
## Not run:
cifti_parcellate(
  "data.dtseries.nii",
  "atlas.dlabel.nii",
  direction = "COLUMN",
  cifti_out = "parcellated.ptseries.nii"
)

## End(Not run)
```

cifti_reduce

Reduce a CIFTI file along a dimension

Description

Wraps `wb_command -cifti-reduce` to perform a reduction operation (e.g. mean, stdev) along a specified direction.

Usage

```
cifti_reduce(
  cifti_in,
  direction = c("ROW", "COLUMN"),
  operation = "MEAN",
  cifti_out,
  only_numeric = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

<code>cifti_in</code>	Path to the input CIFTI file.
<code>direction</code>	Direction to reduce. One of "ROW" or "COLUMN".
<code>operation</code>	Reduction operation. One of "MAX", "MIN", "INDEXMAX", "INDEXMIN", "SUM", "PRODUCT", "MEAN", "STDEV", "SAMPSTDEV", "VARIANCE", "TSNR", "COV", "L2NORM", "MEDIAN", "MODE", "COUNT_NONZERO".
<code>cifti_out</code>	Path for the output CIFTI file.
<code>only_numeric</code>	Logical; exclude non-numeric values?
<code>verbose</code>	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-reduce")` in a session with `wb_command` available.

Examples

```
## Not run:
cifti_reduce(
  "data.dtseries.nii",
  direction = "ROW",
  operation = "MEAN",
  cifti_out = "mean.dscalar.nii"
)

## End(Not run)
```

<code>cifti_stats</code>	<i>Compute statistics on a CIFTI file</i>
--------------------------	---

Description

Wraps `wb_command -cifti-stats` to compute statistics along CIFTI columns.

Usage

```
cifti_stats(
  cifti_in,
  operation = NULL,
  percentile = NULL,
  column = NULL,
  roi = NULL,
  show_map_name = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

<code>cifti_in</code>	Path to the input CIFTI file.
<code>operation</code>	Reduction operation. One of "MAX", "MIN", "INDEXMAX", "INDEXMIN", "SUM", "PRODUCT", "MEAN", "STDEV", "SAMPSTDEV", "VARIANCE", "TSNR", "COV", "L2NORM", "MEDIAN", "MODE", "COUNT_NONZERO".
<code>percentile</code>	Numeric percentile to compute (alternative to operation).
<code>column</code>	Integer; compute for a single column only (1-indexed).
<code>roi</code>	Path to a CIFTI ROI file.
<code>show_map_name</code>	Logical; print map name before each result?
<code>verbose</code>	Logical; print command output?

Value

Character vector of output lines (stats values).

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-stats")` in a session with `wb_command` available.

Examples

```
## Not run:
cifti_stats("data.dscalar.nii", operation = "MEAN")
cifti_stats("data.dscalar.nii", percentile = 95)

## End(Not run)
```

`cifti_weighted_stats` *Compute weighted statistics on a CIFTI file*

Description

Wraps `wb_command -cifti-weighted-stats` to compute spatially-weighted statistics along CIFTI columns.

Usage

```
cifti_weighted_stats(
  cifti_in,
  operation = NULL,
  percentile = NULL,
  spatial_weights = FALSE,
  cifti_weights = NULL,
  column = NULL,
  roi = NULL,
  show_map_name = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

<code>cifti_in</code>	Path to the input CIFTI file.
<code>operation</code>	One of "MEAN", "STDEV", "SAMPSTDEV", "SUM".
<code>percentile</code>	Numeric percentile (alternative to operation).
<code>spatial_weights</code>	Logical; use vertex areas and voxel volumes as weights?

cifti_weights Path to a CIFTI file to use as weights.
 column Integer; compute for a single column only (1-indexed).
 roi Path to a CIFTI ROI file.
 show_map_name Logical; print map name before each result?
 verbose Logical; print command output?

Value

Character vector of output lines (stats values).

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-cifti-weighted-stats")` in a session with `wb_command` available.

Examples

```
## Not run:
cifti_weighted_stats("data.dscalar.nii", operation = "MEAN",
                    spatial_weights = TRUE)

## End(Not run)
```

file_information *Display information about a file*

Description

Wraps `wb_command -file-information` to list information about a data file's content.

Usage

```
file_information(
  data_file,
  only_map_names = FALSE,
  only_number_of_maps = FALSE,
  only_metadata = FALSE,
  only_cifti_xml = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

data_file Path to the input file.
 only_map_names Logical; show only map names?
 only_number_of_maps Logical; show only number of maps?
 only_metadata Logical; show only metadata?
 only_cifti_xml Logical; show only CIFTI XML (CIFTI files only)?
 verbose Logical; print command output?

Value

Character vector of information lines, invisibly.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-file-information")` in a session with `wb_command` available.

Examples

```
## Not run:
file_information("data.dscalar.nii")
file_information("data.dscalar.nii", only_map_names = TRUE)

## End(Not run)
```

<code>get_wb_path</code>	<i>Find the wb_command executable</i>
--------------------------	---------------------------------------

Description

Searches for the `wb_command` binary using R options, environment variables, default installation paths, and `PATH`.

Usage

```
get_wb_path(simplify = TRUE)
```

Arguments

`simplify` If TRUE, return just the path string. If FALSE, return the full settings list.

Value

Character path to `wb_command`, or a settings list when `simplify = FALSE`.

Examples

```
## Not run:  
get_wb_path()  
  
## End(Not run)
```

get_wb_setting	<i>Look up a workbench setting</i>
----------------	------------------------------------

Description

Resolves a setting by checking R options, then environment variables, then a list of default paths.

Usage

```
get_wb_setting(opt_var, env_var, defaults = NULL, is_path = TRUE)
```

Arguments

opt_var	Name of the R option (e.g. "wbcmd.path").
env_var	Name of the environment variable (e.g. "WB_PATH").
defaults	Character vector of fallback values to try.
is_path	If TRUE, checks whether the resolved value exists on disk.

Value

A list with elements value, source, and exists.

Examples

```
get_wb_setting("wbcmd.path", "WB_PATH", defaults = "/usr/bin")
```

get_wb_verbosity	<i>Get the workbench verbosity setting</i>
------------------	--

Description

Get the workbench verbosity setting

Usage

```
get_wb_verbosity()
```

Value

Logical indicating whether verbose output is enabled.

Examples

```
get_wb_verbosity()
```

have_wb

Check if Connectome Workbench is available

Description

Check if Connectome Workbench is available

Usage

```
have_wb()
```

Value

TRUE if wb_command is found and executable.

Examples

```
have_wb()
```

metric_math

Evaluate expression on metric files

Description

Wraps wb_command -metric-math to perform element-wise mathematical operations on metric files at each surface vertex.

Usage

```
metric_math(  
  expression,  
  metric_out,  
  var,  
  fixnan = FALSE,  
  verbose = get_wb_verbosity()  
)
```

Arguments

expression	Math expression string (e.g. "(x - mean) / stdev").
metric_out	Path for the output metric file.
var	Named list of input metric file paths. Names correspond to variable names used in expression.
fixnan	Logical; replace NaN results with 0?
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-metric-math")` in a session with `wb_command` available.

Examples

```
## Not run:
metric_math(
  "abs(x - y)",
  metric_out = "diff.func.gii",
  var = list(x = "post.func.gii", y = "pre.func.gii")
)

## End(Not run)
```

metric_stats

Compute spatial statistics on a metric file

Description

Wraps `wb_command -metric-stats` to compute spatial statistics on a metric file.

Usage

```
metric_stats(
  metric_in,
  operation = NULL,
  percentile = NULL,
  column = NULL,
  roi = NULL,
  show_map_name = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

metric_in	Path to the input metric file.
operation	Reduction operation. One of "MAX", "MIN", "INDEXMAX", "INDEXMIN", "SUM", "PRODUCT", "MEAN", "STDEV", "SAMPSTDEV", "VARIANCE", "TSNR", "COV", "L2NORM", "MEDIAN", "MODE", "COUNT_NONZERO".
percentile	Numeric percentile to compute (alternative to operation).
column	Integer; compute for a single column only (1-indexed).
roi	Path to a metric ROI file.
show_map_name	Logical; print map name before each result?
verbose	Logical; print command output?

Value

Character vector of output lines (stats values).

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-metric-stats")` in a session with `wb_command` available.

Examples

```
## Not run:
metric_stats("thickness.func.gii", operation = "MEAN")

## End(Not run)
```

set_wb_path	<i>Set the path to wb_command</i>
-------------	-----------------------------------

Description

Set the path to `wb_command`

Usage

```
set_wb_path(path, check = TRUE)
```

Arguments

path	Path to the <code>wb_command</code> executable.
check	If TRUE, verify the path exists before setting.

Value

The previous value of the option, invisibly.

Examples

```
## Not run:
set_wb_path("/opt/workbench/bin/wb_command")

## End(Not run)
```

surface_curvature	<i>Compute surface curvature</i>
-------------------	----------------------------------

Description

Wraps `wb_command -surface-curvature` to compute mean and/or Gaussian curvature of a surface.

Usage

```
surface_curvature(
  surface_in,
  mean_out = NULL,
  gauss_out = NULL,
  verbose = get_wb_verbosity()
)
```

Arguments

surface_in	Path to the input surface file.
mean_out	Path for the output mean curvature metric file.
gauss_out	Path for the output Gaussian curvature metric file.
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-surface-curvature")` in a session with `wb_command` available.

Examples

```
## Not run:
surface_curvature(
  "lh.midthickness.surf.gii",
  mean_out = "lh.mean_curv.func.gii"
)

## End(Not run)
```

surface_distortion *Measure distortion between surfaces*

Description

Wraps `wb_command -surface-distortion` to compute distortion between two surfaces on the same mesh.

Usage

```
surface_distortion(
  surface_reference,
  surface_distorted,
  metric_out,
  smooth = NULL,
  caret5_method = FALSE,
  edge_method = FALSE,
  local_affine_method = FALSE,
  verbose = get_wb_verbosity()
)
```

Arguments

surface_reference	Path to the reference surface.
surface_distorted	Path to the distorted surface.
metric_out	Path for the output distortion metric file.
smooth	Smoothing sigma for area distortion (mm). Only used with the default area-based method.
caret5_method	Logical; use the Caret5 method?
edge_method	Logical; use edge-based distortion?
local_affine_method	Logical; use local affine method?
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-surface-distortion")` in a session with `wb_command` available.

Examples

```
## Not run:
surface_distortion(
  "lh.sphere.surf.gii",
  "lh.sphere.reg.surf.gii",
  "lh.distortion.func.gii"
)

## End(Not run)
```

```
surface_generate_inflated
      Generate inflated surface
```

Description

Wraps `wb_command -surface-generate-inflated` to create inflated and very-inflated surfaces from an anatomical surface.

Usage

```
surface_generate_inflated(
  anatomical_surface,
  inflated_out,
  very_inflated_out,
  iterations_scale = NULL,
  verbose = get_wb_verbosity()
)
```

Arguments

<code>anatomical_surface</code>	Path to the input anatomical surface file.
<code>inflated_out</code>	Path for the output inflated surface.
<code>very_inflated_out</code>	Path for the output very-inflated surface.
<code>iterations_scale</code>	Scale factor for inflation iterations. Default is 1.0; use 2.5 for 164k meshes.
<code>verbose</code>	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-surface-generate-inflated")` in a session with `wb_command` available.

Examples

```
## Not run:
surface_generate_inflated(
  "lh.midthickness.surf.gii",
  "lh.inflated.surf.gii",
  "lh.very_inflated.surf.gii"
)

## End(Not run)
```

surface_geodesic_distance

Compute geodesic distance from a vertex

Description

Wraps `wb_command -surface-geodesic-distance` to compute the geodesic distance from a single vertex to all other vertices on a surface.

Usage

```
surface_geodesic_distance(
  surface_in,
  vertex,
  metric_out,
  limit = NULL,
  corrected_areas = NULL,
  verbose = get_wb_verbosity()
)
```

Arguments

<code>surface_in</code>	Path to the input surface file.
<code>vertex</code>	Index of the source vertex (0-indexed).
<code>metric_out</code>	Path for the output metric file.
<code>limit</code>	Maximum geodesic distance in mm. Vertices beyond this distance get value -1.

corrected_areas Path to a metric file of corrected vertex areas (for group-average surfaces).
 verbose Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-surface-geodesic-distance")` in a session with `wb_command` available.

Examples

```
## Not run:
surface_geodesic_distance(
  "lh.midthickness.surf.gii",
  vertex = 0,
  metric_out = "geodesic.func.gii",
  limit = 100
)

## End(Not run)
```

surface_information *Display surface information*

Description

Wraps `wb_command -surface-information` to print summary information about a surface file (vertices, triangles, bounding box, spacing).

Usage

```
surface_information(surface_in, verbose = get_wb_verbosity())
```

Arguments

surface_in Path to the input surface file.
 verbose Logical; print command output?

Value

Character vector of information lines, invisibly.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-surface-information")` in a session with `wb_command` available.

Examples

```
## Not run:
surface_information("lh.midthickness.surf.gii")

## End(Not run)
```

surface_vertex_areas *Compute vertex areas of a surface*

Description

Wraps `wb_command -surface-vertex-areas` to measure the surface area (in mm^2) that each vertex is responsible for.

Usage

```
surface_vertex_areas(surface_in, metric_out, verbose = get_wb_verbosity())
```

Arguments

surface_in	Path to the input surface file.
metric_out	Path for the output metric file.
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-surface-vertex-areas")` in a session with `wb_command` available.

Examples

```
## Not run:
surface_vertex_areas(
  "lh.midthickness.surf.gii",
  "lh.vertex_areas.func.gii"
)

## End(Not run)
```

 volume_math

Evaluate expression on volume files

Description

Wraps `wb_command -volume-math` to perform element-wise mathematical operations on volume files at each voxel.

Usage

```

volume_math(
  expression,
  volume_out,
  var,
  fixnan = FALSE,
  verbose = get_wb_verbosity()
)

```

Arguments

expression	Math expression string (e.g. " $(x - y) / y * 100$ ").
volume_out	Path for the output volume file.
var	Named list of input volume file paths. Names correspond to variable names used in expression.
fixnan	Logical; replace NaN results with 0?
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-volume-math")` in a session with `wb_command` available.

Examples

```

## Not run:
volume_math(
  "clamp(x, 0, 100)",
  volume_out = "clamped.nii.gz",
  var = list(x = "input.nii.gz")
)

## End(Not run)

```

volume_to_surface_mapping
Map volume data to a surface

Description

Wraps `wb_command -volume-to-surface-mapping` to project volume data onto a surface mesh.

Usage

```
volume_to_surface_mapping(  
  volume_in,  
  surface,  
  metric_out,  
  method = c("trilinear", "enclosing", "cubic", "ribbon-constrained", "myelin-style"),  
  inner_surface = NULL,  
  outer_surface = NULL,  
  verbose = get_wb_verbosity()  
)
```

Arguments

volume_in	Path to the input volume file.
surface	Path to the surface to map onto.
metric_out	Path for the output metric file.
method	Mapping method. One of "trilinear", "enclosing", "cubic", "ribbon-constrained", "myelin-style".
inner_surface	Path to the inner surface (required for "ribbon-constrained" and "myelin-style").
outer_surface	Path to the outer surface (required for "ribbon-constrained" and "myelin-style").
verbose	Logical; print command output?

Value

The result of the underlying `wb_cmd()` call.

Connectome Workbench Help

Connectome Workbench is not installed. Run `wb_help("-volume-to-surface-mapping")` in a session with `wb_command` available.

Examples

```
## Not run:
volume_to_surface_mapping(
  "bold.nii.gz",
  "lh.midthickness.surf.gii",
  "lh.bold.func.gii",
  method = "ribbon-constrained",
  inner_surface = "lh.white.surf.gii",
  outer_surface = "lh.pial.surf.gii"
)

## End(Not run)
```

wb_cmd

*Execute a Connectome Workbench command***Description**

Central dispatcher for all `wb_command` operations. Command wrappers call this function to construct, execute, and validate workbench commands.

Usage

```
wb_cmd(
  cmd,
  args = character(),
  ...,
  verbose = get_wb_verbosity(),
  intern = verbose
)
```

Arguments

<code>cmd</code>	The <code>wb_command</code> subcommand (e.g. <code>"-cifti-separate"</code>).
<code>args</code>	Character vector of arguments to pass after <code>cmd</code> .
<code>...</code>	Additional arguments, currently unused.
<code>verbose</code>	Logical; print command output? Defaults to <code>get_wb_verbosity()</code> .
<code>intern</code>	Logical; capture output as character vector? Defaults to the value of <code>verbose</code> .

Value

If `intern = TRUE`, a character vector of command output. Otherwise the exit status (integer).

Examples

```
## Not run:
wb_cmd("-version")
wb_cmd("-cifti-separate", c("input.dscalar.nii", "COLUMN"))

## End(Not run)
```

wb_help

Access wb_command help text

Description

Retrieves the help text for a given `wb_command` subcommand.

Usage

```
wb_help(cmd = NULL)
```

Arguments

cmd	The subcommand to get help for (e.g. <code>"-cifti-separate"</code>). If NULL, shows the top-level help listing all subcommands.
-----	---

Value

Character vector of help text lines, invisibly.

Examples

```
## Not run:
wb_help()
wb_help("-cifti-separate")

## End(Not run)
```

`wb_help_rd`*Render wb_command CLI help as Rd markup*

Description

Called at render time by `\Sexpr` in generated `.Rd` files. Returns valid Rd markup showing CLI help or a fallback message.

Usage

```
wb_help_rd(cmd)
```

Arguments

`cmd` The `wb_command` subcommand (e.g. `"-cifti-average"`).

Value

Character string of Rd markup.

Examples

```
## Not run:  
wb_help_rd("-cifti-average")  
  
## End(Not run)
```

`wb_sitrep`*Situation report for Connectome Workbench*

Description

Prints a diagnostic summary of the workbench environment, including path detection, version, and system information.

Usage

```
wb_sitrep()
```

Value

A list of diagnostic information, invisibly.

Examples

```
## Not run:  
wb_sitrep()  
  
## End(Not run)
```

wb_version	<i>Get Connectome Workbench version</i>
------------	---

Description

Get Connectome Workbench version

Usage

```
wb_version()
```

Value

Character string with the workbench version, or an empty string if unavailable.

Examples

```
## Not run:  
wb_version()  
  
## End(Not run)
```

Index

check_wb_result, 2
cifti_average, 3
cifti_correlation, 4
cifti_dilate, 5
cifti_find_clusters, 7
cifti_gradient, 8
cifti_math, 9
cifti_parcellate, 11
cifti_reduce, 12
cifti_stats, 13
cifti_weighted_stats, 14

file_information, 15

get_wb_path, 16
get_wb_setting, 17
get_wb_verbosity, 17
get_wb_verbosity(), 29

have_wb, 18

metric_math, 18
metric_stats, 19

set_wb_path, 20
surface_curvature, 21
surface_distortion, 22
surface_generate_inflated, 23
surface_geodesic_distance, 24
surface_information, 25
surface_vertex_areas, 26
system(), 3

volume_math, 27
volume_to_surface_mapping, 28

wb_cmd, 29
wb_cmd(), 4–6, 8–12, 19, 21, 23–28
wb_help, 30
wb_help_rd, 31
wb_sitrep, 31
wb_version, 32