Package 'gifti'

May 8, 2026

Type Package

Title Reads in 'Neuroimaging' 'GIFTI' Files with Geometry Information

Version 0.9.0

Description Functions to read in the geometry format under the 'Neuroimaging' 'Informatics' Technology Initiative ('NIfTI'), called 'GIFTI' <https://www.nitrc.org/projects/gifti/>. These files contain surfaces of brain imaging data.

License GPL-2

Imports xml2 (>= 1.1.1), base64enc, R.utils, tools, utils

Suggests rgl, grDevices, testthat, knitr, rmarkdown, covr

BugReports https://github.com/muschellij2/gifti/issues

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.3.3

NeedsCompilation no

Author John Muschelli [aut, cre]

Maintainer John Muschelli <muschellij2@gmail.com>

Repository CRAN

Date/Publication 2026-02-06 13:30:07 UTC

Contents

`convert_binary_datatype` *Convert Binary Data Type*

Description

Converts a data type to the size and `what` for `readBin`, necessary for `Base64Binary` and `GZipBase64Binary` formats

Usage

```
convert_binary_datatype(
  datatype = c("NIFTI_TYPE_UINT8", "NIFTI_TYPE_INT32", "NIFTI_TYPE_UINT32",
    "NIFTI_TYPE_FLOAT32")
)
```

Arguments

`datatype`	data type from GIFTI image

Value

List of length 2: with elements of size and what

Examples

```
convert_binary_datatype()
convert_binary_datatype('NIFTI_TYPE_INT32')
testthat::expect_error(convert_binary_datatype('NIFTI_TYPE_BLAH'))
```

convert_endian *Convert Endian from GIFTI*

Description

Converts Endian format from GIFTI

Usage

```
convert_endian(endian)
```

Arguments

endian	character passed from GIFTI XML

Value

Character string

convert_intent *Convert Intent*

Description

Converts the `intent` field from a GIFTI image to a more standard naming

Usage

```
convert_intent(intent)
```

Arguments

intent	(character) string of intent type

Value

A character string

create_data_matrix *Create Data Matrix with ordering respected*

Description

Create Data Matrix with ordering respected

Usage

```
create_data_matrix(
  data,
  dims,
  ordering = c("RowMajorOrder", "ColumnMajorOrder")
)
```

Arguments

data	Data output from data_decoder
dims	Dimensions of output
ordering	Ordering of the data

Value

Matrix of Values

data_array_attributes *Data Array Attributes*

Description

Parses a list of XML data to get the attributes

Usage

```
data_array_attributes(darray)
```

Arguments

darray	List of xml_nodes from GIFTI data array

Value

data.frame of attributes

data_decoder *Array Data Decoder*

Description

Decodes values from a GIFTI image

Usage

```
data_decoder(
  values,
 encoding = c("ASCII", "Base64Binary", "GZipBase64Binary", "ExternalFileBinary"),
  datatype = NULL,
  endian = c("little", "big", "LittleEndian", "BigEndian"),
  ext_filename = NULL,
  n = NULL
)
```

Arguments

values	text from XML of GIFTI image
encoding	encoding of GIFTI values
datatype	Passed to convert_binary_datatype
endian	Endian to pass in readBin
ext_filename	if encoding = "ExternalFileBinary", then this is the external filename
n	number of values to read. Relevant if encoding = "ExternalFileBinary"

Value

Vector of values

Examples

```
if (have_gifti_test_data(outdir = NULL)) {
   gii_files = download_gifti_data(outdir = NULL)
   L = gifti_list(gii_files[1])
   orig = L$DataArray$Data[[1]]
   encoding = attributes(L$DataArray)$Encoding
   datatype = attributes(L$DataArray)$DataType
   endian = attributes(L$DataArray)$Endian
   vals =  data_decoder(orig, encoding = encoding,
   datatype = datatype, endian = endian)
   enc = data_encoder(vals, encoding = encoding,
   datatype = datatype, endian = endian)
   enc == orig
}
```

data_encoder *Array Data Encoder*

Description

Encodes values for a GIFTI image

Usage

```
data_encoder(
  values,
  encoding = c("ASCII", "Base64Binary", "GZipBase64Binary"),
  datatype = NULL,
  endian = c("little", "big", "LittleEndian", "BigEndian")
)
```

Arguments

values	values to be encoded
encoding	encoding of GIFTI values
datatype	Passed to convert_binary_datatype
endian	Endian to pass in readBin

Value

Single character vector

Examples

```
if (have_gifti_test_data(outdir = NULL)) {
   gii_files = download_gifti_data(outdir = NULL)
   L = gifti_list(gii_files[1])
   orig = L$DataArray$Data[[1]]
   encoding = attributes(L$DataArray)$Encoding
   datatype = attributes(L$DataArray)$DataType
   endian = attributes(L$DataArray)$Endian
   vals =  data_decoder(orig, encoding = encoding,
   datatype = datatype, endian = endian)
   enc = data_encoder(vals, encoding = encoding,
   datatype = datatype, endian = endian)
   enc == orig
}
```

decompress_gii *Decompress Gzipped GIFTI (with extension .gz)*

Description

If a GIFTI file is compressed, as in .gii.gz, this will decompress the file. This has nothing to do with the encoding WITHIN the file

Usage

```
decompress_gii(file)
```

Arguments

file	file name of GIFTI file

Value

Filename of decompressed GIFTI

Examples

```
if (have_gifti_test_data(outdir = NULL)) {
   gii_files = download_gifti_data(outdir = NULL)
   outfile = decompress_gii(gii_files[1])
   print(outfile)
}
```

download_gifti_data *Download GIFTI Test Data*

Description

Downloads GIFTI test data from https://www.nitrc.org/frs/download.php/411/BV_GIFTI_1.3.tar.gz

Usage

```
download_gifti_data(
  outdir = system.file(package = "gifti"),
  overwrite = FALSE,
  ...
)
```

Arguments

outdir	Output directory for test file directory
overwrite	Should files be overwritten if already exist?
...	additional arguments to download.file

Value

Vector of file names

---

gifti_list *Convert GIFTI to List*

Description

Reads in a GIFTI file and coerces it to a list

Usage

```
gifti_list(file)
```

Arguments

file	file name of GIFTI file

Value

List of elements

Examples

```
if (have_gifti_test_data(outdir = NULL)) {
   gii_files = download_gifti_data(outdir = NULL)
   L = gifti_list(gii_files[1])
   orig = L$DataArray$Data[[1]]
   encoding = attributes(L$DataArray)$Encoding
   datatype = attributes(L$DataArray)$DataType
   endian = attributes(L$DataArray)$Endian
   vals =  data_decoder(orig, encoding = encoding,
   datatype = datatype, endian = endian)
   enc = data_encoder(vals, encoding = encoding,
   datatype = datatype, endian = endian)
   enc == orig
}
```

gifti_map_value *Map Values to Triangles from GIFTI*

Description

Takes values and maps them to the correct triangles in space.

Usage

```
gifti_map_value(
  pointset,
  triangle,
  values,
  indices = seq(nrow(pointset)),
  add_one = TRUE
)
```

Arguments

pointset	pointset from GIFTI
triangle	triangles from GIFTI
values	Values to map to the triangles. Same length as indices
indices	indices to place the values, must be in the range of 1 and the number of rows of pointset
add_one	Should 1 be added to the indices for the triangle?

Value

A list of coordinates (in triangles) and the corresponding value mapped to those triangles

have_gifti_test_data *Check Presence of GIFTI Test Data*

Description

Checks if GIFTI test data is downloaded

Usage

```
have_gifti_test_data(outdir = system.file(package = "gifti"))
```

Arguments

outdir	Output directory for test file directory

Value

Logical indicator

Examples

```
have_gifti_test_data(outdir = NULL)
```

`is.gifti` *Test if GIFTI*

Description

Simple wrapper to determine if class is GIFTI

Usage

```
is.gifti(x)

is_gifti(x)
```

Arguments

`x`	object to test

Value

Logical if `x` is GIFTI

`readgii` *Read GIFTI File*

Description

Reads a GIFTI File and parses the output

Usage

```
readgii(file)

readGIfTI(file)

read_gifti(file)
```

Arguments

`file`	Name of file to read

Value

List of values

Examples

```
if (have_gifti_test_data(outdir = NULL)) {
   gii_files = download_gifti_data(outdir = NULL)
   gii_list = lapply(gii_files, readgii)
   surf_files = grep("white[.]surf[.]gii", gii_files, value = TRUE)
   surfs = lapply(surf_files, surf_triangles)

   col_file = grep("white[.]shape[.]gii", gii_files, value = TRUE)
   cdata = readgii(col_file)
   cdata = cdata$data$shape
   mypal = grDevices::colorRampPalette(colors = c("blue", "black", "red"))
   n = 4
   breaks = quantile(cdata)
    ints = cut(cdata, include.lowest = TRUE, breaks = breaks)
    ints = as.integer(ints)
    stopifnot(!any(is.na(ints)))
    cols = mypal(n)[ints]
    cols = cols[surfs[[1]]$triangle]
}
## Not run:
if (have_gifti_test_data(outdir = NULL)) {

 if (requireNamespace("rgl", quietly = TRUE)) {
    rgl::open3d()
    rgl::triangles3d(surfs[[1]]$pointset, color = cols)
    rgl::play3d(rgl::spin3d(), duration = 5)
 }
}

## End(Not run)
```

`surf_triangles` *Make Triangles from GIfTI Image*

Description

Creates Triangles for plotting in RGL from a GIfTI image

Usage

```
surf_triangles(file)
```

Arguments

`file`	File name of GIfTI image, usually `surf.gii`

Value

List of values corresponding to the `data` element from `readgii`

`writegii` *Write .gii xml from "gifti" object*

Description

Writes a "gifti" object to a GIFTI file (ends in *.gii).

Usage

```
writegii(gii, out_file, use_parsed_transformations = FALSE)

writeGIfTI(gii, out_file, use_parsed_transformations = FALSE)

write_gifti(gii, out_file, use_parsed_transformations = FALSE)
```

Arguments

`gii`	The "gifti" object
`out_file`	Where to write the new GIFTI file
`use_parsed_transformations`	Should the `$parsed_transformations` be written instead of the `transformations`? Use if the XML pointers in `transformations` might be invalid. Default: `FALSE`

Index