Package ‘WhiteStripe’

August 30, 2022

Type  Package
Title  White Matter Normalization for Magnetic Resonance Images
Version  2.4.2
Description  Shinohara (2014) <doi:10.1016/j.nicl.2014.08.008> introduced 'WhiteStripe', an intensity-based normalization of T1 and T2 images, where normal appearing white matter performs well, but requires segmentation. This method performs white matter mean and standard deviation estimates on data that has been rigidly-registered to the 'MNI' template and uses histogram-based methods.
License  GPL-3
Depends  R (>= 2.10), methods
Imports  graphics, stats, utils, oro.nifti (>= 0.5.0), mgcv, neurobase
LazyData  true
BugReports  https://github.com/muschellij2/WhiteStripe/issues
Suggests  knitr, rmarkdown
VignetteBuilder  knitr
Encoding  UTF-8
LazyDataCompression  xz
NeedsCompilation  no
Author  R. Taki Shinohara [aut],
        John Muschelli [aut, cre] (<https://orcid.org/0000-0001-6469-1750>)
Maintainer  John Muschelli <muschellij2@gmail.com>
Repository  CRAN
Date/Publication  2022-08-30 08:10:06 UTC

R topics documented:

  download_img_data .................................................. 2
  get.deriv.smooth.hist ........................................... 3
download_img_data

Description

Download T1 and T2 data for Examples

Usage

download_img_data(lib.loc = NULL)

Arguments

lib.loc a character vector with path names of R libraries. Passed to img_data

Value

Logical indicator if the files were downloaded.
get.deriv.smooth.hist

**Description**
This function outputs the nth derivative of a histogram smooth.

**Usage**
```
get.deriv.smooth.hist(x, coefs, knots, deg = 4, deriv.deg = 1)
```

**Arguments**
- `x`: values from smooth_hist
- `coefs`: Coefficients from GAM from smooth_hist
- `knots`: Number of knots fit for GAM
- `deg`: Degree of polynomials
- `deriv.deg`: The degree of the derivative.

**Value**
Derivative of smoothed histogram

**Examples**
```
data(smoothed_histogram)
dy<-get.deriv.smooth.hist(xvals,
coefs=s.hist$coefs,
knots=s.hist$knots,
deg=s.hist$deg,
deriv.deg=1)
```

get.first.mode

**Description**
This function grabs the first peak or shoulder.

**Usage**
```
get.first.mode(x, y, rare.prop = 1/5, verbose = TRUE, remove.tail = TRUE, ...)
```
get.largest.mode

Arguments

- **x**: values of midpoints from `hist`
- **y**: values of counts from `hist`
- **rare.prop**: Proportion used to remove rare intensity tail
- **verbose**: print diagnostic output
- **remove.tail**: Remove rare intensity tail
- **...**: arguments to be passed to `smooth_hist`

Value

Value of `x` that is the first peak

Examples

```r
# Data
data(t1.voi.hist)

# System time
system.time({
  y = t1.voi.hist$counts
  x = t1.voi.hist$mids
  x = x[!is.na(y)];
  y = y[!is.na(y)]
  # 20 used for speed of example
  nawm_peak = get.first.mode(x, y, k=20)
  plot(t1.voi.hist, border="red")
  abline(v=nawm_peak)
})
```

get.largest.mode  *Grab largest peak*

Description

This function grabs the largest peak of the histogram

Usage

```r
get.largest.mode(x, y, verbose = TRUE, ...)
```

Arguments

- **x**: values of midpoints from `hist`
- **y**: values of counts from `hist`
- **verbose**: print diagnostic output
- **...**: arguments to be passed to `smooth_hist`
get.last.mode

Value

Value of x that is the largest peak

Examples

data(t2.voi.hist)
system.time({
y = t2.voi.hist$counts
x = t2.voi.hist$mids
x = x[!is.na(y)];
y = y[!is.na(y)]
# 30 used for speed of example
nawm_peak = get.largest.mode(x, y, k=30)
plot(t2.voi.hist, border="red")
abline(v=nawm_peak)
})

get.last.mode  Get Last Peak

Description

This function grabs the last peak or shoulder.

Usage

get.last.mode(x, y, rare.prop = 1/5, verbose = TRUE, remove.tail = TRUE, ...)

Arguments

x  values of midpoints from hist
y  values of counts from hist
rare.prop  Proportion used to remove rare intensity tail
verbose  print diagnostic output
remove.tail  Remove rare intensity tail
...  arguments to be passed to smooth_hist

Value

Value of x that is the last peak
Examples

```r
data(t1.voi.hist)
system.time{
y = t1.voi.hist$counts
x = t1.voi.hist$mids
x = x[!is.na(y)];
y = y[!is.na(y)]
# 20 used for speed of example
nawm_peak = get.last.mode(x, y, k=20)
plot(t1.voi.hist, border="red")
abline(v=nawm_peak)
}
```

Description

Creates a VOI of Image for the specified slices

Usage

```r
make_img_voi(img, slices = 80:120, na.rm = TRUE, ...)
```

Arguments

- `img`  
  Image (T1 usually or T2). Array or object of class nifti
- `slices`  
  Slices to take for the image voi
- `na.rm`  
  Remove NAs from mean. This is for double checking
- `...`  
  Arguments passed from other methods (not used)

Value

VOI of image.
s.hist

Smoothed histogram of image

Description
Smoothed histogram of image

Usage
s.hist

Format
A GAM from mgcv for x and y from histograms

Examples
## Not run:
data(t2.voi.hist)
y = t2.voi.hist$counts
x = t2.voi.hist$mids
x = x[!is.na(y)];
y = y[!is.na(y)]
# 70 used for speed of example
s.hist = smooth_hist(x, y, k=70)

## End(Not run)

smooth_hist

Histogram smoothing for whitestripe

Description
Uses a generalized additive model (GAM) to smooth a histogram for whitestripe

Usage
smooth_hist(
  x,
  y,
  deg = 4,
  k = floor(min(250, length(x)/2)),
  method = "REML",
  ...
)

t1.voi.hist

Arguments

- **x**: values of midpoints from `hist`
- **y**: values of counts from `hist`
- **deg**: degree of polynomials used
- **k**: Number of knots
- **method**: Method for smoothing for GAM
- **...**: Arguments passed to `gam`

Value

List of objects: x and y coordinates of histogram, coefficients from GAM, fitted values from GAM, the GAM model, the knots fitted, and degrees of polynomials

See Also

`gam`

Examples

```r
data(t2.voi.hist)
y = t2.voi.hist$counts
x = t2.voi.hist$mids
x = x[!is.na(y)];
y = y[!is.na(y)]
# 30 used for speed of example
s.hist = smooth_hist(x, y, k=30)
plot(t2.voi.hist, border="red")
lines(s.hist)
```

t1.voi.hist  
*Histogram of VOI of T1 template image*

Description

Histogram of VOI of T1 template image

Usage

t1.voi.hist

Format

A volume of interest histogram from a T1 image for smoothing
t2.voi.hist

Examples

```
## Not run:
lib.loc = tempdir()
if (download_img_data(lib.loc = lib.loc)){
  t1 = readNIfTI(system.file("T1Strip.nii.gz", package="WhiteStripe",
                        lib.loc = lib.loc))
  t1.voi = make_img_voi(t1)
  any(is.na(t1.voi))
    # FALSE
  t1.voi.hist = hist(t1.voi,
                    breaks=2000,
                    plot=FALSE)
  #save(t1.voi.hist, file="data/t1.voi.hist.rda", compress = TRUE,
  # compression_level=9)
}
## End(Not run)
```

```
## Not run:
lib.loc = tempdir()
if (download_img_data(lib.loc = lib.loc))(
  t2 = readNIfTI(system.file("T2Strip.nii.gz", package="WhiteStripe",
                       lib.loc = lib.loc))
  t2.voi = make_img_voi(t2)
  any(is.na(t2.voi))
    # FALSE
  t2.voi.hist = hist(t2.voi,
                     breaks=2000,
                     plot=FALSE)
  #save(t2.voi.hist, file="data/t2.voi.hist.rda", compress = TRUE,
  # compression_level=9)
}
## End(Not run)
```

---

### t2.voi.hist

**Histogram of VOI of T2 template image**

**Description**

Histogram of VOI of T2 template image

**Usage**

`t2.voi.hist`

**Format**

A histogram volume of interest from a T2 image for smoothing

**Examples**

```
## Not run:
lib.loc = tempdir()
if (download_img_data(lib.loc = lib.loc))(
  t2 = readNIfTI(system.file("T2Strip.nii.gz", package="WhiteStripe",
                     lib.loc = lib.loc))
  t2.voi = make_img_voi(t2)
  any(is.na(t2.voi))
    # FALSE
  t2.voi.hist = hist(t2.voi,
                      breaks=2000,
                      plot=FALSE)
  #save(t2.voi.hist, file="data/t2.voi.hist.rda", compress = TRUE,
  # compression_level=9)
}
## End(Not run)
```
Performs White Stripe of T1 or T2 Images

Description

Returns the mean/sd of the whitestripe and indices for them on the image

Usage

whitestripe(
  img,
  type = c("T1", "T2", "FA", "MD", "first", "last", "largest"),
  breaks = 2000,
  whitestripe.width = 0.05,
  whitestripe.width.l = whitestripe.width,
  whitestripe.width.u = whitestripe.width,
  arr.ind = FALSE,
  verbose = TRUE,
  stripped = FALSE,
  slices = NULL,
  ...
)

Arguments

img Image (T1, T2, FA, or MD). Array or object of class nifti

type T1, T2, FA, or MD image whitestripe

breaks Number of breaks passed to hist

whitestripe.width Radius of the white stripe

whitestripe.width.l Lower Radius of the white stripe

whitestripe.width.u Upper Radius of the white stripe

arr.ind Whether indices should be array notation or not, passed to which

verbose Print diagnostic information

stripped Applying to skull-stripped image. NOTE: This does NOT do a subset of slices, as make_img_voi.

slices slices to use for make_img_voi if only a subset to estimate the distribution.

... Arguments to be passed to get.last.mode
Details

This function takes in an image and computes a window of the distribution called the white stripe. If you wish to pass in values you have subset, such as single from a skull-stripped image, you can pass in `img` and set the class to `img_voi` (class(`img`) = "img_voi") and this will not rerun `make_img_voi`.

Value

List of indices of whitestripe, last mode of histogram, array/nifti of 0/1 corresponding to the mask, mean of whitestripe, standard deviation of whitestripe

Examples

```r
## Not run:
library(WhiteStripe)
lib.loc = tempdir()
if (WhiteStripe::download_img_data(lib.loc = lib.loc)){
  library(oro.nifti)
  set.seed(1)
  t1 = readNIfTI(system.file("T1Strip.nii.gz", package="WhiteStripe", lib.loc = lib.loc))
  t1.ind = whitestripe(t1, "T1")
  set.seed(2)
  t1_2 = readNIfTI(system.file("T1Strip.nii.gz", package="WhiteStripe", lib.loc = lib.loc))
  t1_2.ind = whitestripe(t1_2, "T1")
  t1.mask = whitestripe_ind_to_mask(t1, t1.ind$whitestripe.ind)
  t1.mask[t1.mask == 0] = NA
  orthographic(t1, t1.mask, col.y="red")
  t2 = readNIfTI(system.file("T2Strip.nii.gz", package="WhiteStripe", lib.loc = lib.loc))
  t2.ind = whitestripe(t2, "T2")
  t2.mask = whitestripe_ind_to_mask(t2, t2.ind$whitestripe.ind)
  t2.mask[t2.mask == 0] = NA
  orthographic(t2, t2.mask, col.y="red")
}
## End(Not run)
```

whitestripe_hybrid

Hybrid WhiteStripe

Description

Uses t1 and t2 WhiteStripe to get an intersection of the two masks for a hybrid approach

Usage

```r
whitestripe_hybrid(t1, t2, ...)
```
whitestripe_ind_to_mask

WhiteStripe Indices to Mask

Description

Uses WhiteStripe indices to create image mask

Usage

whitestripe_ind_to_mask(img, indices, writeimg = FALSE, ...)

Arguments

- **img**: Array or class nifti that is corresponds to dimensions of the images the indices were generated from
- **indices**: indices from whitestripe
- **writeimg**: logical to write image or not
- **...**: arguments to passed to writeNIfTI for writing image

Examples

```r
## Not run:
lib.loc = tempdir()
if (download_img_data(lib.loc = lib.loc)){
  t1 = readNIfTI(system.file("T1Strip.nii.gz", package="WhiteStripe", lib.loc = lib.loc))
  t2 = readNIfTI(system.file("T2Strip.nii.gz", package="WhiteStripe", lib.loc = lib.loc))
  ind = whitestripe_hybrid(t1, t2)
}
## End(Not run)
```
whitestripe_norm

Value

Class of array or nifti depending on img input

See Also

whitestripe, whitestripe_hybrid

Examples

## Not run:
lib.loc = tempdir()

if (download_img_data(lib.loc = lib.loc)){
  t1 = readNIfTI(system.file("T1Strip.nii.gz", package="WhiteStripe", 
                   lib.loc = lib.loc))
  t2 = readNIfTI(system.file("T2Strip.nii.gz", package="WhiteStripe", 
                   lib.loc = lib.loc))
  ind = whitestripe_hybrid(t1, t2)
  mask = whitestripe_ind_to_mask(t1, ind$whitestripe.ind)
  orthographic(mask)
}

## End(Not run)

whitestripe_norm Normalize Image using white stripe

Description

Taking the indices from white stripe to normalize the intensity values of the brain

Usage

whitestripe_norm(img, indices, ...)

Arguments

img Array or object of class nifti
indices Indices of white stripe from whitestripe. Can also be a mask (indices where mask > 0 are used.)
... arguments to be passed to mean and sd

Value

Object of same class as img, but normalized
### ws_img_data

*Return Filenames of T1 and T2 data*

**Description**

Return filenames T1 and T2 data for example and vignettes

**Usage**

```r
ws_img_data(lib.loc = NULL, warn = TRUE)
```

**Arguments**

- `lib.loc`: a character vector with path names of R libraries. Passed to `system.file`
- `warn`: Should a warning be printed if the images were not there

**Value**

Vector of filenames

### xvals

*Midpoints from VOI histogram*

**Description**

Midpoints from VOI histogram

**Usage**

```r
xvals
```

**Format**

- x values from histogram for VOI
Index

* datasets
  s.hist, 7
  t1.voi.hist, 8
  t2.voi.hist, 9
  xvals, 14
* hybrid
  whitestripe_hybrid, 11
  whitestripe_ind_to_mask, 12
* whitestripe
  whitestripe_hybrid, 11
  whitestripe_ind_to_mask, 12

download_img_data, 2

gam, 8
get.deriv.smooth.hist, 3
get.first.mode, 3
get.largest.mode, 4
get.last.mode, 5, 10

hist, 4, 5, 8, 10
hybrid(whitestripe_hybrid), 11

img_data, 2

make_img_voi, 6, 10, 11
mean, 13

s.hist, 7
sd, 13
smooth_hist, 4, 5, 7
system.file, 14

t1.voi.hist, 8
t2.voi.hist, 9

which, 10
whitemask(whitestripe_ind_to_mask), 12
whitestripe, 10, 12, 13
whitestripe_hybrid, 11
whitestripe_ind_to_mask, 12