Package ‘neurobase’

January 14, 2020

Type Package

Title 'Neuroconductor' Base Package with Helper Functions for 'nifti'

Objects

Version 1.29.0

Maintainer John Muschelli <muschellij2@gmail.com>

Description Base package for 'Neuroconductor', which includes many helper functions that interact with objects of class 'nifti', implemented by package 'oro.nifti', for reading/writing and also other manipulation functions.

Imports methods, abind, matrixStats, R.utils, graphics, grDevices, stats, RNifti

Depends oro.nifti (>= 0.9.0), R (>= 3.2.0)

License GPL-2

Suggests testthat, ggplot2, knitr, rmarkdown, dplyr, reshape2, httr, covr

VignetteBuilder knitr

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

Encoding UTF-8

RoxygenNote 7.0.1

NeedsCompilation no

Author John Muschelli [aut, cre]

Repository CRAN

Date/Publication 2020-01-14 16:10:02 UTC

R topics documented:

applyEmptyImageDimensions-methods ........................................ 3
boxplot.nifti ............................................................... 4
breaker ........................................................................... 5
checkimg-methods ................................................................. 6
R topics documented:

checknii-methods .................................................. 7
checknii.gz-methods ............................................. 8
check_mask ......................................................... 9
check_mask_fail .................................................. 9
check_nifti-methods ............................................. 10
check_nifti_header-methods .................................... 12
check_outfile ..................................................... 13
cog ............................................................... 14
colorbar ............................................................ 14
copyNIfTIHeader .................................................. 15
cut.nifti ............................................................ 16
datatype ............................................................. 17
density.nifti ....................................................... 17
dicer ................................................................. 18
double_ortho ...................................................... 19
dropEmptyImageDimensions ..................................... 19
emptyImageDimensionsMask ..................................... 21
ensure_array ....................................................... 22
fast_dice_tab ..................................................... 22
fast_readnii ....................................................... 23
file_imgext ........................................................ 23
finite_img-methods ............................................... 24
flip_img ............................................................. 25
getEmptyImageDimensions ...................................... 25
hist.nifti ........................................................... 26
images2matrix ..................................................... 27
img_colour_df ..................................................... 27
img_indices ......................................................... 28
img_list_to_ts ................................................... 29
img_ts_to_df ....................................................... 29
img_ts_to_list ................................................... 30
img_ts_to_matrix ............................................... 30
maskEmptyImageDimensions-methods ......................... 31
mask_img ........................................................... 32
mask_vals .......................................................... 33
mean.nifti .......................................................... 33
minmax_img-methods ............................................. 34
multi_overlay ..................................................... 35
newnii ............................................................... 37
niftiarr ............................................................ 38
nii.stub ............................................................ 38
ortho2 ............................................................... 39
ortho_diff .......................................................... 41
parse_img_ext ..................................................... 43
quantile.nifti ..................................................... 44
quantile_img ....................................................... 45
randomize_mask ................................................... 45
random_nifti ....................................................... 46
applyEmptyImageDimensions-methods

Apply Subsetting from Empty Image Dimensions

Description

Simple wrapper for subsetting an image with indices, dropping empty dimensions.

Usage

```r
applyEmptyImageDimensions(img, inds, reorient = FALSE, ...)
```

---

Index

63

---

applyEmptyImageDimensions-methods

Apply Subsetting from Empty Image Dimensions

Description

Simple wrapper for subsetting an image with indices, dropping empty dimensions.

Usage

```r
applyEmptyImageDimensions(img, inds, reorient = FALSE, ...)
```
## S4 method for signature 'anlz'
applyEmptyImageDimensions(img, inds, reorient = FALSE, ...)

## S4 method for signature 'ANY'
applyEmptyImageDimensions(img, inds, reorient = FALSE, ...)

apply_empty_dim(img, ...)

### Arguments

- **img**: image, nifti object, or array
- **inds**: indices of subset from `getEmptyImageDimensions` or `dropEmptyImageDimensions`.
- **reorient**: Should image be reoriented if a filename?
- **...**: not used

### Value

Object of class `nifti` or `array` if `nifti` is not supplied

### Note

`apply_empty_dim` is a shorthand for `applyEmptyImageDimensions` with all the same arguments.

### See Also

`getEmptyImageDimensions`, `dropEmptyImageDimensions`

---

**boxplot.nifti**

*Boxplot of Values in an Image*

### Description

Computes the boxplot of values of an image with the option for a mask.

### Usage

```r
## S3 method for class 'nifti'
boxplot(x, ..., mask)

## S3 method for class 'anlz'
boxplot(x, ..., mask)
```

### Arguments

- **x**: Object of class `nifti`
- **...**: Arguments passed to `boxplot.default`
- **mask**: object to subset the image. If missing, then all values of the image are plotted.
**breaker**

**Value**

Output of boxplot

**Examples**

```r
img = nifti(array(rnorm(10^3), dim = rep(10, 3)))
mask = img > 0
boxplot(img, mask = mask)
```

---

**breaker** 

*Find Breaks for nifti Image Plotting*

---

**Description**

Helper function for plotting - returns breaks for image plot function for object of class nifti

**Usage**

```r
breaker(x, zlim, col, breaks = NULL)
```

**Arguments**

- **x**: Object of class nifti
- **zlim**: A user-specified zlim. If NULL, will calculate how ortho2 would calculate zlim
- **col**: colors to be plotted. Only used for length(col), so can be a vector of length cols to be plotted
- **breaks**: if !is.null(breaks), then will calculate breaks. Otherwise will return this breaks vector

**Value**

Vector of length 2

If breaks = NULL, then vector of length(col) + 1, otherwise returns breaks
Description

Ensures the output to be a character filename (or vector) from an input image or nifti.

Usage

```r
checkimg(file, allow_array = FALSE, ...)
## S4 method for signature 'nifti'
checkimg(file, allow_array = FALSE, ...)
## S4 method for signature 'ANY'
checkimg(file, allow_array = FALSE, ...)
## S4 method for signature 'character'
checkimg(file, allow_array = FALSE, ...)
## S4 method for signature 'list'
checkimg(file, allow_array = FALSE, ...)
```

Arguments

- `file` character or nifti object
- `allow_array` allow arrays to be passed in
- `...` options passed to `tempimg`

Value

character filename of image or temporary nii, with .nii extension

Author(s)

John Muschelli <muschellij2@gmail.com>
checknii-methods

---

**checknii-methods**  
*Force object to filename with .nii extension*

---

**Description**

Ensures the output to be a character filename (or vector) from an input image or nifti, but not gzipped and has .nii extension.

**Usage**

```r
checknii(file, ...)
```

## S4 method for signature 'nifti'

```r
checknii(file, ...)
```

## S4 method for signature 'character'

```r
checknii(file, ...)
```

## S4 method for signature 'list'

```r
checknii(file, ...)
```

## S4 method for signature 'ANY'

```r
checknii(file, ...)
```

```r
ensure_nii(file, ...)
```

**Arguments**

- `file` character or nifti object
- `...` options passed to `checkimg`

**Value**

character filename of image or temporary nii, with .nii extension

**Author(s)**

John Muschelli <muschellij2@gmail.com>
checkniigz-methods  Force object to filename with .nii.gz extension

Description
Ensures the output to be a character filename (or vector) from an input image or nifti, to be gzipped and has .nii.gz extension

Usage
checkniigz(file, ...)

## S4 method for signature 'nifti'
checkniigz(file, ...)

## S4 method for signature 'ANY'
checkniigz(file, ...)

## S4 method for signature 'character'
checkniigz(file, ...)

## S4 method for signature 'list'
checkniigz(file, ...)

ensure_nii_gz(file, ...)

Arguments
file  character or nifti object
...
options passed to checkimg

Value
Character filename of image or temporary nii, with .nii.gz extension

Author(s)
John Muschelli <muschelli2@gmail.com>
check_mask

Check Mask is Binary

Description

Determine if only values in a mask are 0/1

Usage

check_mask(mask, allow.NA = FALSE, allow.array = TRUE)

Arguments

mask Object of class nifti
allow.NA allow NAs in the mask
allow.array if class(mask) is "array", is this OK?

Value

Logical indicating if object is binary mask with only 0, 1, and NA if applicable

Examples

arr = array(rbinom(1000, size = 1, prob = 0.2), dim = c(10,10,10))
nim = oro.nifti::nifti(arr)
check_mask(nim)

check_mask_fail

Check Mask is Binary. Fail otherwise

Description

Determine if only values in a mask are 0/1. Will error otherwise.

Usage

check_mask_fail(...)
Examples

```r
arr = array(rbinom(1000, size = 1, prob = 0.2), dim = c(10,10,10))
nim = oro.nifti::nifti(arr)
check_mask_fail(nim)
```

Description

Simple check to see if input is character or of class nifti

Usage

```r
check_nifti(
  x,
  reorient = FALSE,
  allow.array = FALSE,
  fast = FALSE,
  need_header = TRUE,
  ...
)
```

## S4 method for signature 'nifti'
```r
check_nifti(
  x,
  reorient = FALSE,
  allow.array = FALSE,
  fast = FALSE,
  need_header = TRUE,
  ...
)
```

## S4 method for signature 'character'
```r
check_nifti(
  x,
  reorient = FALSE,
  allow.array = FALSE,
  fast = FALSE,
  need_header = TRUE,
  ...
)
```

## S4 method for signature 'factor'
```r
check_nifti(
  x,
  reorient = FALSE,
  allow.array = FALSE,
  fast = FALSE,
  need_header = TRUE,
check_nifti-methods

    allow.array = FALSE,
    fast = FALSE,
    need_header = TRUE,
...
)

## S4 method for signature 'list'
check_nifti(
    x,
    reorient = FALSE,
    allow.array = FALSE,
    fast = FALSE,
    need_header = TRUE,
...)

## S4 method for signature 'array'
check_nifti(
    x,
    reorient = FALSE,
    allow.array = FALSE,
    fast = FALSE,
    need_header = FALSE,
...
)

## S4 method for signature 'anlz'
check_nifti(
    x,
    reorient = FALSE,
    allow.array = FALSE,
    fast = FALSE,
    need_header = TRUE,
...)

## S4 method for signature 'ANY'
check_nifti(
    x,
    reorient = FALSE,
    allow.array = FALSE,
    fast = FALSE,
    need_header = TRUE,
...)

Arguments

x character path of image or an object of class nifti, or array
reorient (logical) passed to readnii if the image is to be re-oriented
allow.array (logical) Are array types allowed (TRUE) or should there be an error if the object is not character or class nifti.
fast if TRUE, then fast_readnii will be used versus readnii if the files need to be read in.
need_header if TRUE, then an image type with header information will be returned. If not, then an array is fine. Used really only in conjunction with allow.array
... additional arguments to pass to readnii if relevant

Value
nifti object or array if allow.array=TRUE and x is an array

Author(s)
John Muschelli <muschellij2@gmail.com>

See Also
readnii

Examples
x = nifti()
check_nifti(x)

check_nifti_header-methods

Check if nifti image or read in a nifti header

Description
Simple check to see if input is character or of class nifti and read in the header

Usage
check_nifti_header(x)

## S4 method for signature 'nifti'
check_nifti_header(x)

## S4 method for signature 'character'
check_nifti_header(x)

## S4 method for signature 'factor'
check_nifti_header(x)
## S4 method for signature 'list'
check_nifti_header(x)

## S4 method for signature 'array'
check_nifti_header(x)

## S4 method for signature 'anlz'
check_nifti_header(x)

## S4 method for signature 'ANY'
check_nifti_header(x)

**Arguments**

- **x**
  character path of image or an object of class nifti, or array

**Value**

- nifti object or character

**Author(s)**

John Muschelli <muschellij2@gmail.com>

---

### check_outfile

**Check output filename**

**Description**

This function checks if an output filename is not NULL in conjunction whether the user would like to return an image

**Usage**

`check_outfile(outfile, retimg, fileext = "\.nii.gz")`

**Arguments**

- **outfile**
  output filename or NULL
- **retimg**
  Should an image be returned
- **fileext**
  a non-empty character vector giving the file extension

**Value**

Filename of output file or a temporary filename
cog

Image Center of Gravity

Description
Find Center of Gravity of Image, after thresholding

Usage
cog(img, thresh = 0, ceil = FALSE, warn = TRUE)

Arguments
- **img**: Object of class nifti
- **thresh**: threshold for image, will find img > 0
- **ceil**: Run ceiling to force integers (usu for plotting)
- **warn**: Produce a warning if the image is empty after thresholding

Value
Vector of length 3

Examples
dims = rep(20, 3)
x = array(rnorm(prod(dims)), dim = dims)
img = nifti(x, dim=dims,
datatype = convert.datatype()$FLOAT32, cal.min = min(x),
cal.max = max(x), pixdim = rep(1, 4))
cog(img)

colorbar

Add a colorbar to an ortho object

Description
Adds a series of colors mapped to a value

Usage
colorbar(breaks, col, text.col = "white", labels = TRUE, maxleft = 0.95)
copyNIfTIHeader

Arguments

breaks a set of finite numeric breakpoints for the colours (see image
col a list of colors (see image
text.col axis and text label color
labels labels for tick marks - see axis
maxleft Extent the left hand for colorbar

Value

A plot

Note

Much of this was taken from vertical.image.legend from the aqfig package

copyNIfTIHeader Copy NIfTI Header to an array

Description

Copies slots of a nifti object to an array. This is useful if you're subsetting 4D data and getting an array out

Usage

copyNIfTIHeader(
    img,
    arr,
    drop_slots = c(".Data", "dim_"),
    drop = TRUE,
    onlylast = TRUE,
    warn = TRUE,
    ...
)

Arguments

img object of class nifti to copy header
arr array to copy header information
drop_slots Slots not to copy over from header
drop Should dropImageDimension be called before returning?
onlylast if drop = TRUE, passed to dropImageDimension, if only the last dimensions should be dropped
warn if drop = TRUE, passed to dropImageDimension, for warning print out
... arguments to pass to nifti
Value

Object of class nifti the size of arr

Examples

```r
ing = nifti(img = array(rnorm(10^4), dim=rep(10, 4), dim=rep(10, 4), datatype = 16)
sub = img[,,,1:3]
copyNIfTIHeader(img, sub)
sub = img[,,,1, drop=FALSE]
copyNIfTIHeader(img, sub)
copyNIfTIHeader(img, sub, drop = FALSE)
```

---

**cut.nifti**

Perform Cut on an image

Description

Cuts a numeric image into an integer factor, with the option of a mask.

Usage

```r
## S3 method for class 'nifti'
cut(x, breaks, ..., mask)
## S3 method for class 'anlz'
cut(x, ..., mask)
```

Arguments

- `x` Object of class nifti
- `breaks` either a numeric vector of two or more unique cut points or a single number (greater than or equal to 2) giving the number of intervals into which x is to be cut. Passed to `cut`
- `...` Arguments passed to `cut`
- `mask` object to subset the image. If missing, then all values of the image are used

Value

Object of class nifti with an attribute of levels

Examples

```r
img = nifti(array(rnorm(10^3), dim = rep(10, 3)))
mask = img > 0
cut(img, mask = mask, breaks = 4)
```
**datatype**

*Change Data type for img*

**Description**

Tries to figure out the correct datatype for image. Useful for image masks - makes them binary if

**Usage**

```
datatypes(
    img,
    type_string = NULL,
    datatype = NULL,
    bitpix = NULL,
    trybyte = TRUE,
    warn = TRUE
)
```

**Arguments**

- `img`: nifti object (or character of filename)
- `type_string`: (NULL) character of datatype and bitpix. Supercedes both datatype and bitpix. If specified `convert.datatype[[type_string]]` and `convert.bitpix[[type_string]]` will be used.
- `datatype`: (NULL) character of datatype see `convert.datatype`
- `bitpix`: (NULL) character of bitpix see `convert.bitpix`
- `trybyte`: (logical) Should you try to make a byte (UINT8) if image in c(0, 1)?
- `warn`: Should a warning be issued if defaulting to FLOAT32?

**Value**

object of type nifti

---

**density.nifti**

*Density of Values in an Image*

**Description**

Computes the density of values of an image with the option for a mask.
Usage

## S3 method for class 'nifti'
 density(x, ..., mask)

## S3 method for class 'anlz'
 density(x, ..., mask)

Arguments

x Object of class nifti
...
Arguments passed to density.default
mask object to subset the image. If missing, then all values of the image are plotted.

Value

Output of density

Examples

img = nifti(array(rnorm(10^3), dim = rep(10, 3)))
mask = img > 0
density(img, mask = mask)

dicer

Calculate Dice from a Table

Description

Simple wrapper to calculate the Dice Coefficient/Similarity Index from a table

Usage

dicer(tab, verbose = TRUE)

Arguments

tab table or matrix that is 2 by 2
verbose should the Dice be printed before returned?

Value

Numeric scalar (one number)

Examples

tab = matrix(c(1000, 20, 20, 400), ncol = 2)
dicer(tab)
**double_ortho**

*Double Orthographic Display*

**Description**

Copy of oro.nifti's `orthographic` function with some tweaks such as adding L/R designations for left and right.

**Usage**

```r
double_ortho(
  x,
  y = NULL,
  col.y = gray(0:64/64),
  NA.x = TRUE,
  mfrow = c(2, 4),
  add = FALSE,
  ...
)
```

**Arguments**

- `x` is an object of class nifti or similar.
- `y` is an object of class nifti or similar to be set aside `x`.
- `col.y` is grayscale (by default).
- `NA.x` Set any values of 0 in `x` to NA
- `mfrow` (numeric) layout of the 3 slices
- `add` Should the `y`-plot be added or its own plot? Used in `double_ortho`
- `...` other arguments to `ortho2`

**See Also**

- `orthographic`

---

**dropEmptyImageDimensions**

*Drop Empty Image Dimensions*

**Description**

Drops dimensions of an image that has all irrelevant values.
Usage

dropEmptyImageDimensions(
  img,
  value = 0,
  threshold = 0,
  other.imgs = NULL,
  keep_ind = FALSE,
  reorient = FALSE
)

drop_empty_dim(
  img,
  value = 0,
  threshold = 0,
  other.imgs = NULL,
  keep_ind = FALSE,
  reorient = FALSE
)

Arguments

img nifti object
value Value to check against. If zero, then dropEmptyImageDimensions will drop any
dimension that has fewer than threshold zeroes. May be a vector of values,
matched with match
threshold Drop dimension if fewer than threshold voxels are in the slice
other.imgs List of other nifti objects or filenames to apply the same dropping as img.
keep_ind keep indices in output. Will return list, even if other.imgs not specified
reorient Should image be reoriented if a filename?

Value

List of output image indices, and other images if other.imgs not specified or keep_ind = TRUE.
Otherwise object of class nifti

Note

drop_empty_dim is a shorthand for dropEmptyImageDimensions with all the same arguments.
Also, NA are set to zero.

See Also

gEmptyImageDimensions
Examples

```r
set.seed(5)
dims = rep(10, 3)
arr = array(rnorm(prod(dims)), dim = dims)
arr[,,10] = 0
nim = oro.nifti::nifti(arr)

dnim = dropEmptyImageDimensions(nim, keep_ind = TRUE)
new_nim = dnim$outimg
names(dnim)
```

emptyImageDimensionsMask

`Make Mask from Empty Image Dimensions`

Description

Make a mask of an image that has all irrelevant values

Usage

```r
emptyImageDimensionsMask(img, ..., reorient = FALSE)
empty_dim_mask(img, ..., reorient = FALSE)
```

Arguments

- `img` nifti object
- `...` Arguments to be passed to `getEmptyImageDimensions`
- `reorient` Should image be reoriented if a filename?

Value

Object of class nifti, with binary values

Note

`empty_dim_mask` is a shorthand for `emptyImageDimensionsMask` with all the same arguments.

See Also

`getEmptyImageDimensions`
ensure_array

Ensure an array output

Description
Forces an array output for manipulation and overall conversion

Usage
ensure_array(img)

Arguments
img Image object (nifti or niftiImage)

Value
Array of same dimensions as image object

fast_dice_tab

Fast Dice Tabulation

Description
Fast Dice Tabulation

Usage
fast_dice_tab(x, y)
fast_dice(x, y, verbose = FALSE)

Arguments
x A nifti image, filename, or niftiImage
y A nifti image, filename, or niftiImage
verbose A logical indicating output

Value
A table object
Examples

library(oro.nifti)
set.seed(20161007)
dims = rep(10, 3)
arr = array(rnorm(10*10*10), dim = dims)
nim = oro.nifti::nifti(arr) > -1
fast_dice_tab(nim, nim)
fast_dice(nim, nim) == 1

fast_readnii

Reading NIfTI images through RNifti

Description

This function calls the readNifti function from the RNifti package, and then converts the image to a nifti object

Usage

fast_readnii(fname, dtype = TRUE, drop_dim = TRUE)

Arguments

fname  file name of the NIfTI file.
dtype  Should datatyper be run after reading?
drop_dim  Should drop_img_dim be run after reading?

Value

A nifti object

file_imgext

Get Image file extension

Description

Get the image file extension, either .nii, .hdr, .nii.gz, or .hdr.gz

Usage

file_imgext(file, withdot = TRUE)

Arguments

file  Vector of character filenames
withdot  Should the extension begin with "."?
Value

Vector of extensions. If withdot = FALSE, then will return nii instead of .nii

Description

Simple wrapper for setting non-finite values to zero

Usage

finite_img(img, replace = 0)

## S4 method for signature 'nifti'
finite_img(img, replace = 0)

## S4 method for signature 'array'
finite_img(img, replace = 0)

## S4 method for signature 'ANY'
finite_img(img, replace = 0)

## S4 method for signature 'character'
finite_img(img, replace = 0)

## S4 method for signature 'list'
finite_img(img, replace = 0)

Arguments

img character path of image or an object of class nifti, or list of images
replace Value to replace non-finite values to

Value

nifti object

Author(s)

John Muschelli <muschellij2@gmail.com>
flip_img

Flip NIfTI Image

Description
This image will flip x, y, or z direction

Usage
flip_img(img, x = FALSE, y = FALSE, z = FALSE, ...)

Arguments

img  nifti object or character filename
x    (logical) Flip x direction
y    (logical) Flip y direction
z    (logical) Flip z direction
...  Arguments passed to check_nifti

Value
Object of class nifti

Examples

img = random_nifti(rep(15, 3))
flipped = flip_img(img, x = TRUE)
img = random_nifti(rep(15, 2))
flipped = flip_img(img, x = TRUE)

getEmptyImageDimensions

Get Empty Image Dimensions

Description
Creates a list of indices of an image that has all irrelevant values

Usage

getEmptyImageDimensions(img, value = 0, threshold = 0, reorient = FALSE, ...)

get_empty_dim(img, value = 0, threshold = 0, reorient = FALSE, ...)
Arguments

- **img**: nifti object or array
- **value**: Value to check against. If zero, then `getEmptyImageDimensions` will include any dimension that has fewer than `threshold` zeroes. May be a vector of values, matched with `match`
- **threshold**: Include dimension if fewer than `threshold` voxels are in the slice
- **reorient**: Should image be reoriented if a filename?
- **...**: additional arguments to pass to `check_nifti`

Value

List of length 3 of indices.

Note

`get_empty_dim` is a shorthand for `getEmptyImageDimensions` with all the same arguments. Also, NA are set to zero.

---

**hist.nifti**  
*Histogram of Values in an Image*

Description

Computes and displays a histogram of the values of an image with the option for a mask.

Usage

```r
## S3 method for class 'nifti'
hist(x, ..., mask)
```

```r
## S3 method for class 'anlz'
hist(x, ..., mask)
```

Arguments

- **x**: Object of class nifti
- **...**: Arguments passed to `hist.default`
- **mask**: object to subset the image. If missing, then all values of the image are plotted.

Value

Output of `hist`

Examples

```r
img = nifti(array(rnorm(10^3), dim = rep(10, 3)))
mask = img > 0
hist(img, mask = mask)
```
**images2matrix**

*Transform set of images to matrix*

**Description**

Creates a matrix, where the voxels are on the rows and images are on the columns.

**Usage**

```r
images2matrix(imgs, mask = NULL)
```

**Arguments**

- `imgs`: Vector of files or list of images (niftiImage, array, or nifti)
- `mask`: Binary image to subset the voxels

**Value**

Matrix of V by p, where V is the product of the dimensions of one image or the number of voxels in the mask, and p is the number of images.

---

**img_colour_df**

*Convert Image to Data.frame with Colors*

**Description**

Takes in an image and a color scheme, converts that image into a data.frame with the data and a color mapping.

**Usage**

```r
img_colour_df(img, zlim = NULL, breaks = NULL, col = gray(0:64/64))
```

**Arguments**

- `img`: an object to be coerced to nifti using `check_nifti`
- `zlim`: Limits for the domain of the intensities
- `breaks`: Breaks for the intensities to map to colors
- `col`: Colors to map intensities
- `...`: not used
Value

A data.frame with the first columns being the x,y,z (maybe t) coordinates (named `dim` and the dimension number), a value column that contains the intensity information, and a `colour` column representing the color that voxel maps to.

Note

`img_color_df` is a duplicate of `img_colour_df`.

Examples

```r
ing = nifti(array(rnorm(10^3), dim = rep(10, 3)))
df = img_colour_df(img)
```

---

**img_indices**

**Retrieve Image Indices**

Description

Extract image xyz indices (in voxels or millimeters), with the option to append the values.

Usage

```r
img_indices(img, mask = NULL, add_values = FALSE, units = c("index", "mm"))
```

Arguments

- `img` Object of class `nifti`
- `mask` Mask to be applied for indices the index
- `add_values` Should the value be column-bound to the matrix
- `units` Should the indices be in xyz-coordinates or millimeters.

Value

Matrix of 3 columns if `add_values = FALSE` or 4 columns, otherwise.
**img_list_to_ts**  
*Image List to Time Series*

**Description**

Turns a list of 3D images into a 4D time series image

**Usage**

```
img_list_to_ts(imgs, copy_nifti = TRUE, warn = TRUE)
```

**Arguments**

- **imgs**: object of class `list`, each with 3 dimensions,
- **copy_nifti**: Should a `nifti` object be returned (TRUE) or a simply array (FALSE). Should only be used for slight speed up when array is adequate
- **warn**: Should a warning be printed if object is not class `nifti`

**Value**

Object of class `nifti`

**Note**

If the object is not of class `list`, then the object is returned

---

**img_ts_to_df**  
*Image Time Series to Data.frame*

**Description**

Turns a 4D time series image to a Data.frame

**Usage**

```
img_ts_to_df(imgs, warn = FALSE)
```

**Arguments**

- **imgs**: object of class `nifti` with 4 dimensions, aka a 4D time series
- **warn**: Should a warning be printed if object is not class `nifti` (e.g. a list instead)

**Value**

Matrix of values
**img_ts_to_list**  
*Image Time Series to list*

**Description**

Turns a 4D time series image to a list of 3D images

**Usage**

```r
img_ts_to_list(imgs, copy_nifti = TRUE, warn = TRUE)
```

**Arguments**

- `imgs` object of class `nifti` with 4 dimensions, aka a 4D time series
- `copy_nifti` Should `nifti` objects be returned (TRUE) or simply arrays (FALSE). Should only be used for slight speed up when array is adequate
- `warn` Should a warning be printed if object is not class `nifti`

**Value**

List of images

**Note**

If the object is not of class `nifti` or have 4 dimensions, then the object is returned

---

**img_ts_to_matrix**  
*Image Time Series to Matrix*

**Description**

Turns a 4D time series image to a Matrix

**Usage**

```r
img_ts_to_matrix(imgs, warn = FALSE)
```

**Arguments**

- `imgs` object of class `nifti` with 4 dimensions, aka a 4D time series
- `warn` Should a warning be printed if object is not class `nifti` (e.g. a list instead)

**Value**

Matrix of values
Description

Simple wrapper for replacing indices with a value

Usage

```r
maskEmptyImageDimensions(img, inds, reorient = FALSE, mask.value = 0, ...)
```

### S4 method for signature 'nifti'

```r
maskEmptyImageDimensions(img, inds, reorient = FALSE, mask.value = 0, ...)
```

### S4 method for signature 'character'

```r
maskEmptyImageDimensions(img, inds, reorient = FALSE, mask.value = 0, ...)
```

### S4 method for signature 'factor'

```r
maskEmptyImageDimensions(img, inds, reorient = FALSE, mask.value = 0, ...)
```

### S4 method for signature 'list'

```r
maskEmptyImageDimensions(img, inds, reorient = FALSE, mask.value = 0, ...)
```

### S4 method for signature 'array'

```r
maskEmptyImageDimensions(img, inds, reorient = FALSE, mask.value = 0, ...)
```

### S4 method for signature 'anlz'

```r
maskEmptyImageDimensions(img, inds, reorient = FALSE, mask.value = 0, ...)
```

### S4 method for signature 'ANY'

```r
maskEmptyImageDimensions(img, inds, reorient = FALSE, mask.value = 0, ...)
```

mask_empty_dim(img, ...)

Arguments

- `img` image, nifti object, or array
- `inds` indices of subset from `getEmptyImageDimensions` or `dropEmptyImageDimensions`
- `reorient` Should image be reoriented if a filename?
- `mask.value` Value to replace voxels outside the mask.
- `...` not used

Value

Object of class `nifti` or `array` if `nifti` is not supplied
**Note**

`mask_empty_dim` is a shorthand for `maskEmptyImageDimensions` with all the same arguments.

**See Also**

`getEmptyImageDimensions`, `dropEmptyImageDimensions`

---

### `mask_img`

**Mask Image**

**Description**

Takes an image, masks it by `mask`, and returns an object of class `nifti`.

**Usage**

```r
mask_img(img, mask, allow.NA = TRUE)
```

**Arguments**

- `img` object of class `nifti`
- `mask` array or object of class `nifti`, same dimensions as `img`
- `allow.NA` allow NAs in the mask

**Value**

Object of class `nifti` with values outside mask set to 0 if mask is 0 and NA if mask is NA and `img` if mask == 1

**Examples**

```r
set.seed(5)
dims = rep(10, 3)
arr = array(rnorm(prod(dims)), dim = dims)
nim = oro.nifti::nifti(arr)
mask = abs(nim) > 1
masked = mask_img(nim, mask)
mask[mask == 0] = NA
na_masked = mask_img(nim, mask, allow.NA = TRUE)
```
**Extract or Replace Values Inside a Mask**

**Description**
Methods that act on the `.Data` field in a NIfTI/ANALYZE image but only on values inside a mask.

**Usage**

```r
mask_vals(object, mask)

mask_vals(object, mask) <- value
```

**Arguments**

- `object` is an object of class `nifti` or `anlz`.
- `mask` is an object of class `nifti` or `anlz`.
- `value` is the value to assign to the `.Data` field.

**Examples**

```r
img = nifti(array(rnorm(10^3), dim = rep(10, 3)))
mask = img > 1.5
mask_vals(img, mask)
mask_vals(img, mask) = rep(4, sum(mask))
mask_vals(as(img, "array"),
          as(mask, "array")) = rep(4, sum(mask))
```

**Mean of Values in an Image**

**Description**
Computes the mean of values of an image with the option for a mask.
Usage

```r
## S3 method for class 'nifti'
mean(x, ..., mask)

## S3 method for class 'anlz'
mean(x, ..., mask)
```

Arguments

- `x`: Object of class `nifti`
- `...`: Arguments passed to `mean.default`
- `mask`: Object to subset the image. If missing, then all values of the image are plotted.

Value

Output of `mean`

Examples

```r
img = nifti(array(rnorm(10^3), dim = rep(10, 3)))
mask = img > 0
mean(img, mask = mask)
```

minmax_img-methods Normalize Image using Range

Description

Calculates the range of values in an image, then scales the image minimum to 0 and maximum to 1.

Usage

```r
minmax_img(img)
```

```r
## S4 method for signature 'nifti'
minmax_img(img)

## S4 method for signature 'array'
minmax_img(img)

## S4 method for signature 'ANY'
minmax_img(img)

## S4 method for signature 'character'
minmax_img(img)

## S4 method for signature 'list'
minmax_img(img)
```
multi_overlay

Arguments

img character path of image or an object of class nifti, or list of images

Value

A nifti object (or list of them) or class of object passed in if not specified

multi_overlay Create Multi-Image Plot with Overlays

Description

Creates a multi-row or multi-column plot with image slices and the potential for overlays as well.

Usage

multi_overlay(
  x,
  y = NULL,
  z = NULL,
  w = 1,
  mask = NULL,
  col.x = gray(0:64/64),
  col.y = hotmetal(),
  zlim.x = NULL,
  zlim.y = NULL,
  ybreaks = NULL,
  plane = c("axial", "coronal", "sagittal"),
  xlab = "",
  ylab = "",
  axes = FALSE,
  direction = c("horizontal", "vertical"),
  par.opts = list(oma = c(0, 0, 0, 0), mar = rep(0, 4), bg = "black"),
  text = NULL,
  text.x = 0.5,
  text.y = 1.4,
  text.cex = 2.5,
  text.col = "white",
  main = NULL,
  main.col = text.col,
  main.cex = text.cex,
  NA.x = TRUE,
  NA.y = TRUE,
  pdim = NULL,
  useRaster = TRUE,
  ...
)
multi_overlay_center(x, y = NULL, ...)

Arguments

x        List of images of class nifti or character vector of filenames
y        List of images of class nifti or character vector of filenames. Same length as x.
z        Slice to display.
w        3D volume to display if x has 4-D elements
mask     nifti image to drop empty image dimensions if wanted. Passed to dropEmptyImageDimensions
col.x     Color to display x images
col.y     Color to display y images
zlim.x    Limits for x to plot
zlim.y    Limits for y to plot
ybreaks   (numeric) breaks for y to passed to image
plane     the plane of acquisition to be displayed
xlab      Label for x-axis
ylab      Label for y-axis
axes      Should axes be displayed
direction Should images be a row or column? Ignored if mfrow is in par.opts
par.opts  Options to pass to par
text      Text to be displayed
text.x    Location of text in x-domain
text.y    Location of text in y-domain
text.cex  Multiplier for text font
text.col  Color for text and main.
main      Title for each plot
main.col  Color for main. Will default to text.col
main.cex  Multiplier for text font. Will default to text.cex
NA.x      Should 0's in x be set to NA?
NA.y      Should 0's in y be set to NA?
pdim      Pixel dimensions if passing in arrays. Will be overridden if x is a nifti object
useRaster if TRUE, a bitmap raster is used to plot the image instead of polygons. Passed to image
...      Additional arguments to pass to image
Examples

## Not run:

```r
if (require(brainR)) {
  visits = 1:3
  y = paste0("Visit_{}, visits, ".nii.gz")
  y = system.file(y, package = "brainR")
  y = lapply(y, readnii)

  y = lapply(y, function(r){
    pixdim(r) = c(0, rep(1, 3), rep(0, 4))
    dropImageDimension(r)
  })

  x = system.file("MNI152_T1_1mm_brain.nii.gz",
                package = "brainR")
  x = readnii(x)
  mask = x >0
  x = lapply(visits, function(tmp){
    x
  })
  alpha = function(col, alpha = 1) {
    cols = t(col2rgb(col, alpha = FALSE)/255)
    rgb(cols, alpha = alpha)
  }
  multi_overlay(x, y,
                col.y = alpha(hotmetal(), 0.5),
                mask = mask,
                main = paste0("\n", "Visit_{}, visits),
                text = LETTERS[visits],
                text.x = 0.9,
                text.y = 0.1,
                text.cex = 3)
}
```

## End(Not run)

---

**newnii**

Resets image parameters for a copied nifti object

Description

Resets the slots of a nifti object, usually because an image was loaded, then copied and filled in with new data instead of making a nifti object from scratch. Just a wrapper for smaller functions

Usage

```r
newnii(img, ...)
```
Arguments

img  nifti object (or character of filename)
...
arguments to be passed to datatype

Value

object of type nifti

---

**niftiarr**  
*Make new nifti from array*

Description

Make new nifti object by passing in old nifti and array

Usage

niftiarr(img, arr)

Arguments

img  object of class nifti
arr  array to be passed in to .Data slot

Value

object of class nifti

---

**nii.stub**  
*Grab nii file stubname*

Description

Quick helper function to strip off .nii or .nii.gz from filename

Usage

nii.stub(x, bn = FALSE)

Arguments

x  character vector of filenames ending in .nii or .nii.gz
bn  Take basename of file?

Value

A character vector with the same length as x
orth2

Orthographic Display, added options

Description

Copy of oro.nifti’s orthographic function with some tweaks such as adding L/R designations for left and right

Usage

```r
ortho2(
x,
y = NULL,
xyz = NULL,
w = 1,
col = gray(0:64/64),
col.y = oro.nifti::hotmetal(),
zlim = NULL,
zlim.y = NULL,
NA.x = FALSE,
NA.y = TRUE,
crosshairs = TRUE,
col.crosshairs = "red",
xlab = "",
ylab = "",
axes = FALSE,
oma = c(0, 0, 0, ifelse(ycolorbar, 5, 0)),
mar = rep(0, 4),
bg = "black",
text = NULL,
text.color = "white",
text.cex = 2,
text.x = 32,
text.y = 32,
add.orient = FALSE,
mfrow = c(2, 2),
ybreaks = NULL,
breaks = NULL,
addlegend = FALSE,
leg.x = 32,
leg.y = 32,
legend,
leg.col,
leg.title = NULL,
leg.cex,
window = NULL,
ycolorbar = FALSE,
)```


clables = TRUE,
add = TRUE,
pdim = NULL,
useRaster = is.null(y),
mask = NULL,
...
)

Arguments

x is an object of class nifti or similar.
y is an object of class nifti or similar for the overlay.
xyz is the coordinate for the center of the crosshairs.
w is the time point to be displayed (4D arrays only).
col is grayscale (by default).
col.y is hotmetal (by default).
zlim is the minimum and maximum ‘z’ values passed into image.
zlim.y is the minimum and maximum ‘z’ values passed into image for the overlay.
NA.x Set any values of 0 in x to NA
NA.y Set any values of 0 in y to NA
crosshairs is a logical value for the presence of crosshairs in all three orthogonal planes (default = TRUE).
col.crosshairs is the color of the crosshairs (default = red).
xlab is set to "" since all margins are set to zero.
ylab is set to "" since all margins are set to zero.
axes is set to FALSE since all margins are set to zero.
oma is the size of the outer margins in the par function.
mar is the number of lines of margin in the par function.
bg is the background color in the par function.
text allows the user to specify text to appear in the fourth (unused) pane.
text.color is the color of the user-specified text (default = “white”).
text.cex is the size of the user-specified text (default = 2).
text.x x coordinate for text
text.y y coordinate for text
add.orient (logical) Add left/right, A/P, etc. orientation
mfrow (numeric) layout of the 3 slices
ybreaks (numeric) breaks for y to passed to image
breaks (numeric) breaks for x to passed to image
addlegend (logical) add legend?
leg.x (numeric) x coordinate for legend
**ortho_diff**

Plot differences for Prediction and Gold Standard

**Description**

Uses `ortho2` to plot differences between a predicted binary image and the assumed ground truth (`roi`).
Usage

```r
ortho_diff(
  img,
  pred,
  roi,
  xyz = NULL,
  cols = c("#56B4E9", "#D55E00", "#009E73"),
  levels = c("False Negative", "False Positive", "True Positive"),
  addlegend = TRUE,
  center = TRUE,
  leg.cex = 1.5,
  ...
)
```

```r
multi_overlay_diff(
  x,
  pred,
  roi,
  z = NULL,
  cols = c("#56B4E9", "#D55E00", "#009E73"),
  ...
)
```

Arguments

- `img` image to be underlaid
- `pred` binary segmentation (prediction)
- `roi` binary manual segmentation (ground truth)
- `xyz` coordinate for the center of the crosshairs.
- `cols` colors for false negatives/positives
- `levels` labels for false negatives/positives
- `addlegend` add legend, passed to `ortho2`
- `center` run `xyz` on `roi`. Disregarded if `xyz` is not NULL
- `leg.cex` multiplier for legend size
- `...` arguments to be passed to `ortho2` or `multi_overlay`
- `x` List of images of class `nifti` or character vector of filenames
- `z` slice to display

See Also

- `ortho2`
Examples

```r
## Not run:

if (require(brainR)) {
  visits = 1:3
  y = paste0("Visit_", visits, ".nii.gz")
  y = system.file(y, package = "brainR")
  y = lapply(y, readnii)

  y = lapply(y, function(r){
    pixdim(r) = c(0, rep(1, 3), rep(0, 4))
    dropImageDimension(r)
  })

  x = system.file("MNI152_T1_1mm_brain.nii.gz", package = "brainR")
  x = readnii(x)
  mask = x > 0
  alpha = function(col, alpha = 1) {
    cols = t(col2rgb(col, alpha = FALSE)/255)
    rgb(cols, alpha = alpha)
  }
  roi = y[[2]]
  pred = y
  multi_overlay_diff(x, roi = roi, pred = pred)
  multi_overlay_diff(x, roi = roi, pred = pred,
    mask = mask,
    main = paste0("\n", "Visit ", visits),
    text = LETTERS[visits],
    text.x = 0.9,
    text.y = 0.1,
    text.cex = 3)
}
```

## End(Not run)

---

**parse_img_ext**  Parse Image Extensions

### Description

Get image extensions from a filename

### Usage

```r
parse_img_ext(file)
```

### Arguments

- **file**  Character filename to parse
Value

   Extension of file

Examples

   parse_img_ext("blah.nii.gz")
   parse_img_ext("blah.mnc")
   parse_img_ext("blah.nii")
   parse_img_ext("blah")
   parse_img_ext("blah.img")
   parse_img_ext("blah.hdr")
   parse_img_ext("blah.hdr.gz")

quantile.nifti       Sample Quantiles

Description

   Computes sample quantiles for an image, with the option of a mask.

Usage

   ## S3 method for class 'nifti'
   quantile(x, ..., mask)

   ## S3 method for class 'anlz'
   quantile(x, ..., mask)

Arguments

   x          Object of class nifti
   ...        Arguments passed to quantile
   mask       object to subset the image. If missing, then all values of the image are used

Value

   Output of quantile

Examples

   img = nifti(array(rnorm(10^3), dim = rep(10, 3)))
   mask = img > 0
   quantile(img, mask = mask)
**quantile_img**

*Create Quantile Image*

---

**Description**

Creates output image of the quantiles that each voxel is in, after applying the mask.

**Usage**

```r
quantile_img(img, mask = NULL, ...)
```

**Arguments**

- `img`: Character vector, or object of class `nifti`.
- `mask`: Mask to determine cumulative distribution function (cdf) from.
- `...`: Additional arguments to pass to `ecdf`.

**Value**

Object of class `nifti`.

---

**randomize_mask**

*Randomize Image based on Mask*

---

**Description**

Randomize the values within a mask.

**Usage**

```r
randomize_mask(img, mask)
```

**Arguments**

- `img`: Object of class `nifti` with values to be randomized.
- `mask`: Binary mask indicating which values should be randomized.

**Value**

Object of class `nifti`.
Examples

```r
set.seed(5)
dims = rep(10, 3)
arr = array(rnorm(prod(dims)), dim = dims)
nim = oro.nifti::nifti(arr)
mask = abs(nim) > 1
randomize_mask(nim, mask)
```

---

**random_nifti**

*Create Random ‘nifti’ object*

**Description**

Create Random ‘nifti’ object

**Usage**

```r
random_nifti(dim, ...)
```

**Arguments**

- `dim` dimensions for the ‘nifti’ object
- `...` arguments to send to `nifti`

**Value**

A ‘nifti’ object

**Examples**

```r
random_nifti(c(10, 10, 2))
random_nifti(c(10, 10))
```

---

**readNIftI2**

*readNIftI with default non-reorientation*

**Description**

This function calls the `readNIftI` function from the `oro.nifti` package, but sets the reorientation to `FALSE` by default
**Usage**

```r
readNIfTI2(..., reorient = FALSE)
readnii(...,
reorient = FALSE,
dtype = TRUE,
drop_dim = TRUE,
reset_slope = FALSE,
warn = FALSE,
rm_extensions = TRUE)
```

**Arguments**

- `...`: Arguments to pass to `readNIfTI`
- `reorient`: Reorientation argument to pass to `readNIfTI`
- `dtype`: Should `datatyper` be run after reading?
- `drop_dim`: Should `drop_img_dim` be run after reading?
- `reset_slope`: Reset slope/intercept of image
- `warn`: Should warnings from `readNIfTI` be printed? If not, `suppressWarnings` is called. Also passed to `datatyper`
- `rm_extensions`: Should niftiExtensions be converted to simple nifti objects?

**Value**

nifti object

---

**remake_img**  
*Remake Image from Vector*

**Description**

Wrapper function to take a vector of values and result in a nifti object

**Usage**

```r
remake_img(vec, img, mask = NULL, warn = FALSE, ...)
```

**Arguments**

- `vec`: vector of values to be in resulting image
- `img`: object of class nifti to put vector into
- `mask`: binary array/ nifti object to denote where vector values are to be.
- `warn`: Should a warning be issued if defaulting to FLOAT32?
- `...`: additional arguments passed to `datatyper`
Value

Object of class \texttt{nifti}

See Also

\texttt{niftiarr}

\begin{tabular}{ll}
\textbf{remap\_filename} & \textit{Build Filename (usually for images)} \\
\end{tabular}

Description

This is a simple function that helps with the case where you want to construct a filename (usually for an image) with the same base of the filename, the same directory (default), but things added to the front or end of that base filename, with the same extension.

Usage

\texttt{remap\_filename(x, sub\_dir = NULL, prefix = "", suffix = ")

Arguments

\begin{description}
\item \texttt{x} input filename/character vector
\item \texttt{sub\_dir} sub-directory for the new filename. If \texttt{NULL}, then the directory is the same directory as \texttt{x}
\item \texttt{prefix} string to put in front of base of filename
\item \texttt{suffix} string to put at the end of base of filename
\end{description}

Value

Character vector

Examples

\begin{verbatim}
fname = file.path("/path/to/file", "original.nii.gz")
remap_filename(fname, prefix = "preproc_", ".with.gz")
fname = "original.nii"
remap_filename(fname, prefix = "note_", ".has_directory")
remap_filename(c(fname, "other.nii.gz"), prefix = "note_", ".has_directory")
\end{verbatim}
replace_dropped_dimensions

Remake Dropped Dimensions

Description

This function is the reverse of dropEmptyImageDimensions. If dropEmptyImageDimensions was run, and the output is a list, usually if keep_ind = TRUE, this function reverses that.

Usage

replace_dropped_dimensions(img, inds, orig.dim)

Arguments

- **img**: Object of class nifti where image dimensions were dropped.
- **inds**: List of length 3 of indices from dropEmptyImageDimensions or getEmptyImageDimensions
- **orig.dim**: Original dimension of pre-dropped image. Output image will have dimensions same as this value

Value

Object of class nifti

Examples

```r
## Not run:
# nim is an object of class nifti
dd = dropEmptyImageDimensions(nim, keep_ind = TRUE)
remake = replace_dropped_dimensions(img = dd$outimg,
               inds = dd$inds,
               orig.dim = dd$orig.dim)
all.equal(nim, remake)
## End(Not run)
```

replace_outside_surface

Replace Values Outside Surface of image

Description

Determines values outside the surface of an image and gives a mask back with those values set to a replacement.
Usage

replace_outside_surface(
  img,
  value = 0,  
  threshold = 0,
  replace_value = NA,
  reorient = FALSE
)

Arguments

img nifti object or array
value Value to check against. If zero, then replace_outside_surface will include any dimension that has fewer than threshold zeroes. May be a vector of values, matched with match
threshold Include dimension if fewer than threshold voxels are in the slice
replace_value Value to replace those outside the surface.
reorient Should image be reoriented if a filename? Passed to check_nifti

Value

Creates an array of binary values. If img is a nifti object, then a nifti is returned

rescale_img Image Rescaler

Description

Rescales an image to be in certain value range. This was created as sometimes DICOM scale and slope parameters may be inconsistent across sites and the data need to be value restricted

Usage

rescale_img(
  filename,
  pngname = NULL,
  write.nii = FALSE,
  outfile = NULL,
  min.val = -1024,
  max.val = 3071,
  ROIformat = FALSE,
  drop_dim = TRUE,
  writer = "dcm2nii",
...
)
robust_window

Arguments

- **filename**: filename of image to be read into R or nifti object
- **pngname**: filename of png of histogram of values of image to be made. For no png - set to NULL (default)
- **write.nii**: logical - should the image be written.
- **outfile**: if write.nii = TRUE, filename of output file
- **min.val**: minimum value of image (default -1024 (for CT)). If no thresholding set to -Inf
- **max.val**: maximum value of image (default 3071 (for CT)). If no thresholding set to Inf
- **ROIformat**: if TRUE, any values $< 0$ will be set to 0
- **drop_dim**: Should drop_img_dim be applied?
- **writer**: character value to add to description slot of NIFTI header
- **...**: extra methods to be passed to writenii

Value

Object of class nifti

---

**robust_window**  
*Window image based on quantiles of Image*

Description

Takes an image, finds the quantiles given by `probs`, then sets values outside these values to other values, as determined by `replace` argument passed to `window_img`

Usage

`robust_window(img, non_zero = FALSE, probs = c(0, 0.999), ..., mask = NULL)`

Arguments

- **img**: object of class nifti
- **non_zero**: Should zeroes be excluded from the calculation of quantiles?
- **probs**: quantiles to constrain the image these define the window sent to `window_img`
- **...**: additional arguments sent to `window_img`
- **mask**: binary image to use to to calculate quantiles

Value

Object of class nifti with values outside quantiles replaced by values depending on replace argument passed to `window_img`
same_dims

Description
Wrapper to check if multiple objects all have the same dimensions

Usage
same_dims(...)

Arguments
... Arguments (matrices or arrays) where the dimension will be checked against the first object’s dimension

Value
Logical indicating if all have the same dimensions or not

Examples
mat1 = matrix(1:9, ncol = 3)
mat2 = matrix(rnorm(9), ncol = 3)
mat3 = matrix(rnorm(16), ncol = 4)
mat4 = matrix(rnorm(9), ncol = 3)
same_dims(mat1, mat2)
same_dims(mat1, mat3)
same_dims(mat1, mat2, mat4)

separate_img-methods

Separate Labeled Image into Multiple Binary Images

Description
Takes in an image, gets the unique values, then creates a list of binary images for each one of those values.

Usage
separate_img(img, levels = NULL, drop_zero = TRUE)

## S4 method for signature 'nifti'
separate_img(img, levels = NULL, drop_zero = TRUE)

## S4 method for signature 'array'
slice_colour_df

separate_img(img, levels = NULL, drop_zero = TRUE)

## S4 method for signature 'ANY'
separate_img(img, levels = NULL, drop_zero = TRUE)

## S4 method for signature 'character'
separate_img(img, levels = NULL, drop_zero = TRUE)

## S4 method for signature 'list'
separate_img(img, levels = NULL, drop_zero = TRUE)

Arguments

img character path of image or an object of class nifti, or list of images
levels if levels is given, then the separation is only done for those levels and not unique values of the image.
drop_zero Should zeroes be dropped from the labels? Zero usually denotes background or non-interesting voxels

Value

A nifti object (or list of them) or class of object passed in if not specified

Note

Exact equalling is using ==

slice_colour_df Slice a Image Color Data.frame

Description

Slices a image color data.frame along the 3 planes (axial, coronal, sagittal) and returns it in a ggplot-ready format for faceting.

Usage

slice_colour_df(img_df, xyz = NULL)

Arguments

img_df an image data.frame, usually from img_colour_df. Must have the columns: dim1, dim2, dim3, colour, and value.
xyz coordinates to slice the data.frame in x, y, and z - domains
Value

A data.frame with x and y coordinates, colour, and intensity values, along with the associated planes that were sliced.

Examples

```r
img = nifti(array(rnorm(10^3), dim = rep(10, 3)))
df = img_colour_df(img)
sliced = slice_colour_df(df, c(5, 5, 4))
```

Description

Subset DTI data based on b-values

Usage

```r
subset_dti(
  img,
  bvals,
  bvecs,
  b_step = 1,
  maximum = Inf,
  shells = NULL,
  verbose = TRUE,
  ...
)
```

## S4 method for signature 'nifti'
```r
subset_dti(
  img,
  bvals,
  bvecs,
  b_step = 1,
  maximum = Inf,
  shells = NULL,
  verbose = TRUE,
  ...
)
```

## S4 method for signature 'ANY'
```r
subset_dti(
  img,
  bvals,
  bvecs,
  b_step = 1,
  maximum = Inf,
  shells = NULL,
  verbose = TRUE,
  ...
)
```
subset_dti-methods

subset_dti(
  img,  
  bvals, 
  bvecs, 
  b_step = 1, 
  maximum = Inf, 
  shells = NULL, 
  verbose = TRUE, 
  ... 
)

## S4 method for signature 'character'
subset_dti(
  img, 
  bvals, 
  bvecs, 
  b_step = 1, 
  maximum = Inf, 
  shells = NULL, 
  verbose = TRUE, 
  ...
)

## S4 method for signature 'list'
subset_dti(
  img, 
  bvals, 
  bvecs, 
  b_step = 1, 
  maximum = Inf, 
  shells = NULL, 
  verbose = TRUE, 
  ...
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>img</td>
<td>character or nifti object</td>
</tr>
<tr>
<td>bvals</td>
<td>filename of b-values (assuming 1 row)</td>
</tr>
<tr>
<td>bvecs</td>
<td>filename of b-vectors (assuming 3 rows)</td>
</tr>
<tr>
<td>b_step</td>
<td>step of b-values to round to</td>
</tr>
<tr>
<td>maximum</td>
<td>maximum b-value threshold</td>
</tr>
<tr>
<td>shells</td>
<td>Shells to keep (after rounding)</td>
</tr>
<tr>
<td>verbose</td>
<td>print diagnostic messages</td>
</tr>
<tr>
<td>...</td>
<td>options passed to checkimg</td>
</tr>
</tbody>
</table>

Value

List of filenames of image, b-values, and b-vectors that were subselected.
Author(s)

John Muschelli <muschellij2@gmail.com>

Examples

```r
## Not run:
img = "~/Downloads/data.nii.gz"
bvals = "~/Downloads/bvals"
bvecs = "~/Downloads/bvals"
verbose = TRUE
b_step = 50
maximum = 1500
shells = NULL
sub = subset_dti(img = img, bvals = bvals, bvecs = bvecs,
maximum = 1500,
b_step = 50)

## End(Not run)
```

Description

Takes in an object of class nifti, writes it to a temp file, appends .nii.gz as `writenii` adds it.

Usage

```r
tempimg(
  nim,
  gzipped = TRUE,
  checknan = TRUE,
  check_type = FALSE,
  warn = FALSE,
  drop_dim = TRUE,
  ...
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>nim</code></td>
<td>object of class nifti</td>
</tr>
<tr>
<td><code>gzipped</code></td>
<td>Should file be gzipped? Passed to <code>writenii</code></td>
</tr>
<tr>
<td><code>checknan</code></td>
<td>Check for NAs or NaNs</td>
</tr>
<tr>
<td><code>check_type</code></td>
<td>Check the datatype for an image. Will run <code>datatyper</code>.</td>
</tr>
<tr>
<td><code>warn</code></td>
<td>Should warnings be displayed if <code>writenii</code> has any? Passed to <code>writenii</code></td>
</tr>
<tr>
<td><code>drop_dim</code></td>
<td>Should <code>drop_img_dim</code> be applied?</td>
</tr>
<tr>
<td><code>...</code></td>
<td>Passed to <code>writenii</code>.</td>
</tr>
</tbody>
</table>
window_img

Value
filename of output nii.gz

Description
Windows an image to min and max values and also changes cal_max and cal_min parameters

Usage

window_img(x,
    window = c(0, 100),
    replace = c("window", "missing", "zero"),
    ...
)

Arguments

- x is either a character name for the image or an object of class nifti
- window numeric of length 2 that gives min and max for window
- replace either "window" if the any values outside of c(min, max) are set to the min or max or "missing" for these to be set to NA
- ... not used

Value
Object of class nifti

See Also
readnii

Examples

```r
set.seed(5)
dims = rep(10, 3)
arr = array(rnorm(prod(dims)), dim = dims)
nim = oro.nifti::nifti(arr)
window_img(nim, window = c(1, 5))
window_img(nim, window = c(1, 5), replace = "missing")
tfile = tempimg(nim)
window_img(tfile)
window_img(as.factor(tfile))
arr = window_img(img_data(nim))
```
writeNIfTI2

DESCRIPTION

This function calls the writeNIfTI function from the oro.nifti package, but makes sure to remove .nii extension and warnings can be suppressed.

USAGE

writeNIfTI2(nim, filename, dtype = FALSE, compression = 9, ...)

writteni(
  nim,
  filename,
  dtype = TRUE,
  drop_dim = TRUE,
  warn = FALSE,
  compression = 9,
  rm_extensions = TRUE,
  ...
)

ARGUMENTS

nim object of class nifti, passed to writeNIfTI
filename path to save the NIfTI file. Suffix will be removed
dtype Should datatyper be run before writing?
compression compression level for gzipped files.
... Additional arguments passed to writeNIfTI
drop_dim Should drop_img_dim be run before writing?
warn Should warnings from writeNIfTI be printed? If not, suppressWarnings is called
rm_extensions should niftiExtensions be converted to simple nifti objects before writing?

VALUE

Nothing
write_nifti

Note
While writeNIfTI2 does not run datatyper as default, writenii does. Additional functionality will be added to writenii likely but will not to writeNIfTI2

---

write_nifti  General NIfTI Writer

Description
Writes out NIfTI files for multiple formats. Currently, for nifti objects and niftiImage objects from RNifti

Usage
write_nifti(nim, filename, ...)

Arguments
nim  Container for NIfTI Image
filename  Filename of image to be written out
...  additional arguments, to be passed to writeNIfti or writenii

Value
Output from NIfTI writer

xyz  Image Center of Gravity Wrapper

Description
Find Center of Gravity of Image, after thresholding and take ceiling (wrapper for cog)

Usage
xyz(...)

Arguments
...  Arguments passed to cog

Value
Vector of length 3

Note
Just a convenience wrapper for cog(ceil=TRUE)
zero_pad

Zero pads an image

Description
This function zero pads an image by a certain number of dimensions, usually for convolution.

Usage
```r
zero_pad(img, kdim, invert = FALSE, pad_value = 0, ...)
```

Arguments
- `img`: Array or class nifti
- `kdim`: Dimensions of kernel
- `invert`: (logical) If FALSE, does zero padding. If TRUE, reverses the process.
- `pad_value`: Value to pad the image with. May use other values, such as -1024 for CT data
- `...`: Options to `copyNIfTIHeader`

Value
Object of class nifti

Examples
```r
kdim = c(3,3,5)
img = array(rnorm(30*30*36), dim = c(30, 30, 36))
pad = zero_pad(img, kdim)
back = zero_pad(pad, kdim, invert=TRUE)
all.equal(back, img)
```

zlimmer

Find Image z-limits

Description
Helper function for plotting - returns zlim for `image` plot function

Usage
```r
zlimmer(x, zlim = NULL, computed_range = NULL)
```
Arguments

- `x`: Object of class `nifti`
- `zlim`: A user-specified `zlim`. If `NULL`, will calculate how `ortho2` would calculate `zlim`
- `computed_range`: If the range of the data was already computed, this can be passed in and will be used if relevant.

Value

- If `zlim = NULL`, then vector of length 2, otherwise returns `zlim`

---

**zscore_img**

*Get Z-score over a margin of an img*

---

Description

Standardizes an image either by the axial, sagittal, or coronal slice or whole image

Usage

```r
zscore_img(
  img,
  mask = NULL,
  margin = NULL,
  centrality = c("mean", "median", "trimmed_mean"),
  variability = c("sd", "iqr_diff", "mad", "maddiff", "iqr", "trimmed_sd"),
  trim = 0.2,
  remove.na = TRUE,
  remove.nan = TRUE,
  remove.inf = TRUE,
  remove.val = 0,
  remask = TRUE
)
```

Arguments

- `img`: character path of image or an object of class `nifti`
- `mask`: character path of mask or an object of class `nifti`
- `margin`: Margin of image to z-score over (NULL - whole brain, 3-Axial, 2-Sagittal, 1-Coronal)
- `centrality`: (character) Measure to center the data, either mean or median
- `variability`: (character) Measure to scale the data
- `trim`: if centrality is `trimmed_mean` or variability is `trimmed_sd`, then the amount of trimming
- `remove.na`: (logical) change NAs to `remove.val`
remove.nan  (logical) change NaN to remove.val
remove.inf  (logical) change Inf to remove.val
remove.val  (logical) value to put the NA/NaN/Inf
remask      (logical) Should the image be remasked after normalizing?

Value
Array of object of class nifti

See Also
aperm

Examples
```r
dim = c(100, 30, 5)
ing = array(rnorm(prod(dim), mean=4, sd=4),
dim=dim)

truth2 = img
for (i in 1:dim(img)[2]) {
  truth2[,i,] = (truth2[,i,]- mean(truth2[,i,]))/sd(truth2[,i,])
}

truth1 = img
for (i in 1:dim(img)[1]) {
  truth1[i,,] = (truth1[i,,]- mean(truth1[i,,]))/sd(truth1[i,,])
}

truth3 = img
for (i in 1:dim(img)[3]) {
  truth3[,,i] = (truth3[,,i]- mean(truth3[,,i]))/sd(truth3[,,i])
}
try3 = zscore_img(img, margin=3)
stopifnot(all.equal(try3, truth3))
try2 = zscore_img(img, margin=2)
stopifnot(all.equal(try2, truth2))
try1 = zscore_img(img, margin=1)
stopifnot(all.equal(try1, truth1))

z = zscore_img(img, margin=NULL)
ztrim = zscore_img(img, margin=NULL, centrality = "trimmed_mean", variability = "trimmed_sd")```
Index

aperm, 62
apply_empty_dim
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions,anlz-method
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions,ANY-method
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions,array-method
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions,character-method
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions,factor-method
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions,list-method
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions,nifti-method
  (applyEmptyImageDimensions-methods), 3
applyEmptyImageDimensions-methods, 3
axis, 15
basename, 38
boxplot, 5
boxplot.anlz (boxplot.nifti), 4
boxplot.default, 4
boxplot.nifti, 4
breaker, 5
ceiling, 14
check_mask, 9, 9
check_mask_fail, 9
check_nifti, 25–27, 50
check_nifti (check_nifti-methods), 10
check_nifti,anlz-method
  (check_nifti-methods), 10
check_nifti,ANY-method
  (check_nifti-methods), 10
check_nifti,array-method
  (check_nifti-methods), 10
check_nifti,character-method
  (check_nifti-methods), 10
check_nifti,factor-method
  (check_nifti-methods), 10
check_nifti,list-method
  (check_nifti-methods), 10
check_nifti,nifti-method
  (check_nifti-methods), 10
check_nifti-methods, 10
check_nifti_header
  (check_nifti_header-methods), 12
check_nifti_header,anlz-method
  (check_nifti_header-methods), 12
check_nifti_header,ANY-method
  (check_nifti_header-methods), 12
check_nifti_header,array-method
  (check_nifti_header-methods), 12
check_nifti_header,character-method
  (check_nifti_header-methods), 12
check_nifti_header,factor-method
  (check_nifti_header-methods), 12
check_nifti_header,list-method
  (check_nifti_header-methods), 12
check_nifti_header, nifti-method
   (check_nifti_header-methods), 12
check_nifti_header-methods, 12
check_outfile, 13
checkimg, 7, 8, 55
checkimg (checkimg-methods), 6
checkimg, ANY-method (checkimg-methods), 6
checkimg, character-method
   (checkimg-methods), 6
checkimg, list-method
   (checkimg-methods), 6
checkimg, nifti-method
   (checkimg-methods), 6
checkimg-methods, 6
checknii (checknii-methods), 7
checknii, ANY-method (checknii-methods), 7
checknii, character-method
   (checknii-methods), 7
checknii, list-method
   (checknii-methods), 7
checknii, nifti-method
   (checknii-methods), 7
checknii-methods, 7
checkniigz (checkniigz-methods), 8
checkniigz, ANY-method
   (checkniigz-methods), 8
checkniigz, character-method
   (checkniigz-methods), 8
checkniigz, list-method
   (checkniigz-methods), 8
checkniigz, nifti-method
   (checkniigz-methods), 8
checkniigz-methods, 8
cog, 14, 59
colorbar, 14, 41
convert.bitpix, 17
convert.datatype, 17
copyNIFTIHeader, 15, 60
cut, 16
cut.anlz (cut.nifti), 16
cut.nifti, 16
datatype, 17, 38
datatype, 23, 47, 56, 58, 59
datatype (datatype), 17
density, 18
density.anlz (density.nifti), 17
density.default, 18
density.nifti, 17
dicer, 18
double_ortho, 19
drop_empty_dim
   (dropEmptyImageDimensions), 19
drop_img_dim, 23, 47, 51, 56, 58
dropEmptyImageDimensions, 4, 19, 31, 32, 36, 49
dropImageDimension, 15
ecdf, 45
empty_dim_mask
   (emptyImageDimensionsMask), 21
emptyImageDimensionsMask, 21
ensure_array, 22
ensure_nii (checknii-methods), 7
ensure_nii_gz (checkniigz-methods), 8
fast_dice (fast_dice_tab), 22
fast_dice_tab, 22
fast_readnii, 12, 23
file_imgext, 23
finite_img (finite_img-methods), 24
finite_img, ANY-method
   (finite_img-methods), 24
finite_img, array-method
   (finite_img-methods), 24
finite_img, character-method
   (finite_img-methods), 24
finite_img, list-method
   (finite_img-methods), 24
finite_img, nifti-method
   (finite_img-methods), 24
finite_img-methods, 24
flip_img, 25
get_empty_dim
   (getEmptyImageDimensions), 25
getEmptyImageDimensions, 4, 20, 21, 25, 31, 32, 49
hist, 26
hist.anlz (hist.nifti), 26
hist.default, 26
hist.nifti, 26
image, 5, 15, 36, 40, 41, 60
INDEX

images2matrix, 27
img_color_df (img_colour_df), 27
img_colour_df, 27, 53
img_indices, 28
img_list_to_ts, 29
img_ts_to_df, 29
img_ts_to_list, 30
img_ts_to_matrix, 30

legend, 41
list, 29

mask_empty_dim
  (maskEmptyImageDimensions-methods), 31
mask_img, 32
mask_vals, 33
mask_vals-methods, (mask_vals), 33
mask_vals<- (mask_vals), 33
mask_vals<-, anlz, ANY, ANY-method 
  (mask_vals), 33
mask_vals<-, anlz-method (mask_vals), 33
mask_vals<-, array, ANY, ANY-method 
  (mask_vals), 33
mask_vals<-, array-method (mask_vals), 33
mask_vals<-, nifti, ANY, ANY-method 
  (mask_vals), 33
mask_vals<-, nifti-method (mask_vals), 33
maskEmptyImageDimensions 
  (maskEmptyImageDimensions-methods), 31
maskEmptyImageDimensions, anlz-method 
  (maskEmptyImageDimensions-methods), 31
maskEmptyImageDimensions, ANY-method 
  (maskEmptyImageDimensions-methods), 31
maskEmptyImageDimensions, array-method 
  (maskEmptyImageDimensions-methods), 31
maskEmptyImageDimensions, character-method 
  (maskEmptyImageDimensions-methods), 31
maskEmptyImageDimensions, factor-method 
  (maskEmptyImageDimensions-methods), 31
maskEmptyImageDimensions, list-method 
  (maskEmptyImageDimensions-methods), 31

mean.anlz (mean.nifti), 33
mean.default, 34

minmax_img (minmax_img-methods), 34
minmax_img, ANY-method
  (minmax_img-methods), 34
minmax_img, array-method
  (minmax_img-methods), 34
minmax_img, character-method 
  (minmax_img-methods), 34
minmax_img, list-method 
  (minmax_img-methods), 34
minmax_img, nifti-method
  (minmax_img-methods), 34
minmax_img-methods, 34
multi_overlay, 35, 42
multi_overlay_center (multi_overlay), 35
multi_overlay_diff (ortho_diff), 41

nnewii, 37
nifti, 15, 22, 29, 30, 46–49
niftiarr, 38, 48
nii.stub, 38

par, 36
parse_img_ext, 43

quantile, 44
quantile.anlz (quantile.nifti), 44
quantile.nifti, 44
quantile_img, 45

random_nifti, 46
randomize_mask, 45
readNIfTI, 46, 47
readNIfTI, 23
readNIfTI2, 46
readnii, 12, 57
readnii (readNIfTI2), 46
remake_img, 47
remap_filename, 48
replace_dropped_dimensions, 49
replace_outside_surface, 49
rescale_img, 50
robust_window, 51

same_dims, 52
separate_img (separate_img-methods), 52
separate_img, ANY-method
(separate_img-methods), 52
separate_img, array-method
(separate_img-methods), 52
separate_img, character-method
(separate_img-methods), 52
separate_img, list-method
(separate_img-methods), 52
separate_img, nifti-method
(separate_img-methods), 52
separate_img-methods, 52
slice_colour_df, 53
subset_dti (subset_dti-methods), 54
subset_dti, ANY-method
(subset_dti-methods), 54
subset_dti, character-method
(subset_dti-methods), 54
subset_dti, list-method
(subset_dti-methods), 54
subset_dti, nifti-method
(subset_dti-methods), 54
subset_dti-methods, 54
suppressWarnings, 47, 58

tempimg, 6, 56

window_img, 51, 57
write_nifti, 59
writeNIfTI, 58
writeNifti, 59
writeNIfTI2, 58
writtenii, 51, 56, 59
writtenii (writeNIfTI2), 58

xyz, 42, 59

zero_pad, 60
zlimmer, 60
zscore_img, 61