Package ‘fslr’

August 25, 2022

Type Package

Title Wrapper Functions for 'FSL' ('FMRIB' Software Library) from Functional MRI of the Brain ('FMRIB')

Version 2.25.2

Maintainer John Muschelli <muschellij2@gmail.com>

Description Wrapper functions that interface with 'FSL' <http://fsl.fmrib.ox.ac.uk/fsl/fslwiki/>, a powerful and commonly-used 'neuroimaging' software, using system commands. The goal is to be able to interface with 'FSL' completely in R, where you pass R objects of class 'nifti', implemented by package 'oro.nifti', and the function executes an 'FSL' command and returns an R object of class 'nifti' if desired.

Imports methods, R.utils, graphics, grDevices, stats, utils

Depends oro.nifti (>= 0.5.0), neurobase (>= 1.32.0), R (>= 3.2.0)

License GPL-3

VignetteBuilder knitr

Suggests knitr, rmarkdown, covr

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

SystemRequirements FSL

Encoding UTF-8

RoxygenNote 7.2.1

NeedsCompilation no

Author John Muschelli [aut, cre] (<https://orcid.org/0000-0001-6469-1750>)

Repository CRAN

Date/Publication 2022-08-25 15:12:35 UTC

R topics documented:

applytopup ................................................................. 6
aux.file-methods ......................................................... 7
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>bitpix-methods</td>
<td>7</td>
</tr>
<tr>
<td>cal.max-methods</td>
<td>7</td>
</tr>
<tr>
<td>cal.min-methods</td>
<td>8</td>
</tr>
<tr>
<td>checkout</td>
<td>8</td>
</tr>
<tr>
<td>check_file</td>
<td>9</td>
</tr>
<tr>
<td>data_type-methods</td>
<td>9</td>
</tr>
<tr>
<td>data_type-methods</td>
<td>10</td>
</tr>
<tr>
<td>dim.-methods</td>
<td>10</td>
</tr>
<tr>
<td>download_fsl</td>
<td>11</td>
</tr>
<tr>
<td>dtifit</td>
<td>11</td>
</tr>
<tr>
<td>eddy</td>
<td>12</td>
</tr>
<tr>
<td>eddy_correct</td>
<td>14</td>
</tr>
<tr>
<td>enforce_form</td>
<td>14</td>
</tr>
<tr>
<td>face_removal_mask</td>
<td>15</td>
</tr>
<tr>
<td>fast</td>
<td>16</td>
</tr>
<tr>
<td>fast.help</td>
<td>17</td>
</tr>
<tr>
<td>flirt</td>
<td>18</td>
</tr>
<tr>
<td>flirt.help</td>
<td>19</td>
</tr>
<tr>
<td>flirt_apply</td>
<td>19</td>
</tr>
<tr>
<td>fnirt</td>
<td>20</td>
</tr>
<tr>
<td>fnirt.help</td>
<td>21</td>
</tr>
<tr>
<td>fnirt_with_affine</td>
<td>21</td>
</tr>
<tr>
<td>fnirt_with_affine_apply</td>
<td>22</td>
</tr>
<tr>
<td>fslabs.help</td>
<td>24</td>
</tr>
<tr>
<td>fslacos.help</td>
<td>24</td>
</tr>
<tr>
<td>fsladd.help</td>
<td>25</td>
</tr>
<tr>
<td>fsland</td>
<td>25</td>
</tr>
<tr>
<td>fslasin.help</td>
<td>26</td>
</tr>
<tr>
<td>fslatan.help</td>
<td>26</td>
</tr>
<tr>
<td>fslbet.help</td>
<td>27</td>
</tr>
<tr>
<td>fslbin.help</td>
<td>28</td>
</tr>
<tr>
<td>fslbinv.help</td>
<td>28</td>
</tr>
<tr>
<td>fslchfiletype</td>
<td>29</td>
</tr>
<tr>
<td>fslchfiletype.help</td>
<td>30</td>
</tr>
<tr>
<td>fslcmd</td>
<td>30</td>
</tr>
<tr>
<td>fslcog</td>
<td>31</td>
</tr>
<tr>
<td>fslcos.help</td>
<td>32</td>
</tr>
<tr>
<td>fslcpgeom</td>
<td>33</td>
</tr>
<tr>
<td>fslcpgeom.help</td>
<td>34</td>
</tr>
<tr>
<td>fsldir</td>
<td>34</td>
</tr>
<tr>
<td>fsldiv.help</td>
<td>35</td>
</tr>
<tr>
<td>fsledge.help</td>
<td>35</td>
</tr>
<tr>
<td>fslentropy</td>
<td>36</td>
</tr>
<tr>
<td>fslepi_reg</td>
<td>36</td>
</tr>
<tr>
<td>fslepiode.help</td>
<td>38</td>
</tr>
<tr>
<td>fslexp.help</td>
<td>38</td>
</tr>
<tr>
<td>fslfill.help</td>
<td>39</td>
</tr>
</tbody>
</table>
R topics documented:

fslfill2 ................................................. 39
fslgetorient ........................................... 40
fslhd ...................................................... 41
fslhd.help .............................................. 41
fslhd.parse ............................................ 42
fslhelp .................................................... 42
fslindex.help ........................................... 43
fslllog.help ............................................ 44
fslmask.help ........................................... 44
fslmaths.help ........................................... 45
fslmax ..................................................... 45
fslmean .................................................... 46
fslmerge.help ......................................... 46
fslmul.help ............................................. 47
fslnan.help ............................................. 47
fslnanm.help .......................................... 48
fslor ...................................................... 49
fslorient ............................................... 50
fslorient.help ........................................ 50
fslorienter ............................................. 51
fslrand.help ........................................... 51
fslrandn.help .......................................... 52
fslrange ................................................. 52
fslrecip.help .......................................... 53
fslrem.help ............................................. 54
fslreorient2std ....................................... 54
fslreorient2std.help ................................. 55
fslrobustfov .......................................... 56
fslrobustfov.help ..................................... 57
fslroi ..................................................... 57
fslsd ....................................................... 58
fsslins .................................................... 59
fsslins.help ............................................ 60
fssliscicetimer ........................................ 60
fslsmooth.help ......................................... 61
fslsmooth_in_mask ..................................... 62
fslsplit ................................................. 63
fslsplit.help ........................................... 64
fslsqrt.help ............................................ 64
fslsqrt.help ............................................ 65
fslstats .................................................. 65
fslstats.help .......................................... 66
fslsub.help ............................................. 66
fslsub2.help ........................................... 67
fslsum ..................................................... 68
fslswapdim.help ........................................ 68
fsltan.help ............................................. 69
fslthresh.help ......................................... 69
R topics documented:

fslval ........................................... 70
fslval.help ................................... 70
fslview ......................................... 71
fslview.help .................................. 72
fslvol .......................................... 72
fslvolume ...................................... 73
fslxor .......................................... 73
fsl_abs .......................................... 74
fsl_acos ....................................... 75
fsl_add .......................................... 76
fsl_anat ........................................ 77
fsl_anat.help .................................. 78
fsl_applywarp ................................... 79
fsl_applywarp.help ............................. 80
fsl_asin ........................................ 80
fsl_atan ........................................ 81
fsl_atlas_dir ................................... 82
fsl_avscale ................................... 82
fsl_bet .......................................... 83
fsl_biascorrect ................................ 84
fsl_bin .......................................... 85
fsl_binv ....................................... 86
fsl_bin_tab ..................................... 87
fsl_cluster ..................................... 87
fsl_cos .......................................... 89
fsl_data_dir ................................... 90
fsl_deface ...................................... 91
fsl_dice ........................................ 92
fsl_dilate ...................................... 92
fsl_div .......................................... 94
fsl_edge ........................................ 95
fsl_erode ....................................... 96
fsl_exp .......................................... 97
fsl_fill .......................................... 98
fsl_index ....................................... 99
fsl_log ......................................... 100
fsl_mask ........................................ 101
fsl_maths ....................................... 102
fsl_merge ....................................... 103
fsl_mul .......................................... 104
fsl_nan .......................................... 105
fsl_nann ....................................... 106
fsl_rand ......................................... 107
fsl_randn ....................................... 108
fsl_recip ....................................... 109
fsl_rem .......................................... 110
fsl_resample ................................... 111
fsl_smooth ..................................... 112
R topics documented:

fsl_smoothest ........................................ 113
fsl_sqr ............................................... 114
fsl_sqrt ............................................. 115
fsl_std_dir .......................................... 116
fsl_sub ............................................. 116
fsl_sub2 ........................................... 117
fsl_swapdim ......................................... 118
fsl_tan ............................................. 119
fsl_thresh .......................................... 120
fsl_tsplot .......................................... 122
fsl_version ......................................... 123
get.fsl ............................................. 124
get.fsloutput ...................................... 125
get.imgext .......................................... 125
getForms ............................................ 126
get_quickshear_mask ................................ 126
have.fsl ............................................ 127
intent_code-methods ............................... 128
intent_name-methods ............................... 128
intent_p1-methods ................................. 129
intent_p2-methods ................................. 129
intent_p3-methods ................................. 129
invert_xfm ......................................... 130
magic-methods ..................................... 131
mcflirt .............................................. 131
mcflirt.help ....................................... 132
melodic ............................................... 132
melodic.help ....................................... 133
mid_sagittal_align .................................. 134
mni_fname .......................................... 134
mni_img ............................................. 135
mridefacer ......................................... 136
parse_avscale ...................................... 137
pixdim-methods .................................... 137
probtrackx ......................................... 138
qform.character-method ......................... 140
qform_code-methods ............................... 141
readrpi ............................................. 141
read_xfm ............................................ 142
reverse_rpi_orient .................................. 142
rpi_orient .......................................... 143
run_first_all ...................................... 144
run_first_all.help .................................. 145
scl_inter-methods .................................. 146
scl_slope-methods .................................. 146
sform_code-methods ............................... 146
sizeof_hdr-methods ............................... 147
slice_code-methods ................................ 147
applytopup

Description

A tool for applying and correcting estimated susceptibility induced distortions

Usage

applytopup(
  infile,
  datain,
  index,
  topup_files,
  out = NULL,
  method = c("lsr", "jac"),
  interp = c("spline", "trilinear"),
  verbose = TRUE
)

apply_topup(...)
fsl_applytopup(...)

Arguments

infile list of names of input image (to be corrected)
datain name of text file with PE directions/times
index list of indices into –datain of the input image (to be corrected)
topup_files name of field/movements (from topup)
out basename for output (warped) image
method Use jacobian modulation (jac) or least-squares resampling (lsr). default=lsr.
interp Image interpolation model, trilinear or spline. Default spline
verbose Print diagnostic information while running
... arguments passed to topup if using fsl_topup
aux.file-methods  

Description

aux_file method for character types

Usage

```r
## S4 method for signature 'character'
aux.file(object)
```

Arguments

- `object` is a filename to pass to `fslval`

---

bitpix-methods  

Description

bitpix method for character types

Usage

```r
## S4 method for signature 'character'
bitpix(object)
```

Arguments

- `object` is a filename to pass to `fslval`

---

cal.max-methods  

Description

cal_max method for character types

Usage

```r
## S4 method for signature 'character'
cal.max(object)
```

Arguments

- `object` is a filename to pass to `fslval`
### Description

*cal.min* method for character types

### Usage

```r
## S4 method for signature 'character'
cal.min(object)
```

### Arguments

- `object` is a filename to pass to `fslval`

### Description

This function determines if the determinants of the sform and qform have the same sign

### Usage

```r
checkout(hd)
```

### Arguments

- `hd` (list) sforms from `getForms`

### Value

logical indicating if sform and qform consistent

### Examples

```r
if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard", "MNI152_T1_2mm.nii.gz")
  forms = getForms(mnifile)
  checkout(forms)
}
```
check_file  

Wrapper for getForms with filename

Description
Checking the q/s-forms for a header

Usage
check_file(file, ...)

Arguments
- file (character) filename of image to be checked
- ... options passed to checkimg

Value
result of checkout

Examples
library(fslr)
if (have.fsl()){
  mnifile = mni_fname("2")
  check_file(mnifile)
}

datatype-methods  

Extract Image datatype attribute

Description
datatype method for character types

Usage
## S4 method for signature 'character'
datatype(object)

Arguments
- object is a filename to pass to fslval
**data_type-methods**  
*Extract Image data_type attribute*

**Description**  
data_type method for character types

**Usage**  
```r  
## S4 method for signature 'character'  
data_type(object)  
```

**Arguments**  
- `object` is a filename to pass to `fslval`  

**descrip-methods**  
*Extract Image descrip attribute*

**Description**  
descrip method for character types

**Usage**  
```r  
## S4 method for signature 'character'  
descrip(object)  
```

**Arguments**  
- `object` is a filename to pass to `fslval`  

**dim_-methods**  
*Extract Image dim_ attribute*

**Description**  
dim_ method for class character

**Usage**  
```r  
## S4 method for signature 'character'  
dim_(object)  
```

**Arguments**  
- `object` is a filename to pass to `fslval`
**download_fsl**  
*Download FSL*

**Description**

Download FSL Tarball

**Usage**

```r
download_fsl(
  os = c("macosx", "redhat5", "redhat6", "centos5", "centos6", "debian", "ubuntu"),
  outdir = tempdir(),
  overwrite = TRUE,
  ...
)
```

**Arguments**

- `os` Operating system
- `outdir` Output directory for tarball
- `overwrite` If file.path(outdir, tarball_name) exists, should it be overwritten?
- `...` Arguments to pass to `download.file`

**Value**

Filename of destination file

---

**dtifit**  
*DTI Fitting Procedure from FSL*

**Description**

Calls dtifit from FSL

**Usage**

```r
dtifit(
  infile, 
  bvecs, 
  bvals, 
  mask = NULL, 
  outprefix = NULL, 
  opts = "", 
  bet.opts = "", 
  verbose = TRUE,
  ...
)
```
Arguments

infile  
Input filename

bvecs  
b-vectors: matrix of 3 columns or filename of ASCII text file

bvals  
b-values: vector of same length as number of rows of b-vectors or filename of ASCII text file

mask  
Mask filename

outprefix  
Output prefix

opts  
Additional options for dtifit

bet.opts  
Options for fslbet if mask is not supplied

verbose  
print diagnostic messages

sse  
Save sum of squared errors

save_tensor  
Save tensor file out

grad_image  
Gradient Nonlinearity Tensor file

Value

Vector of character filenames of output. See Note

Note

On successful completion of the command, the following files will be output, which are: mask - the mask used in the analysis outprefix_V1 - 1st eigenvector outprefix_V2 - 2nd eigenvector outprefix_V3 - 3rd eigenvector outprefix_L1 - 1st eigenvalue outprefix_L2 - 2nd eigenvalue outprefix_L3 - 3rd eigenvalue outprefix_MD - mean diffusivity outprefix_FA - fractional anisotropy outprefix_MO - mode of the anisotropy (oblate ~ -1; isotropic ~ 0; prolate ~ 1) outprefix_S0 - raw T2 signal with no diffusion weighting optional output If sse = TRUE, then the additional file will be present: outprefix_sse - Sum of squared error If save_tensor = TRUE, then the additional file will be present: outprefix_tensor - tensor as a 4D file in this order: Dxx,Dxy,Dxz,Dyy,Dyz,Dzz

eddy  
Eddy Current Correction

Description

This function calls eddy from FSL for DTI Processing
Usage

eddy(
  infile,
  mask,
  acq_file,
  index_file,
  bvecs,
  bvals,
  topup = NULL,
  outfile = NULL,
  retimg = TRUE,
  opts = "",
  verbose = TRUE,
  eddy_cmd = c("eddy", "eddy_openmp", "eddy_cuda"),
  ...
)

Arguments

infile input filename of 4D image.
mask Mask filename (or class nifti)
acq_file A text-file describing the acquisition parameters for the different images in infile
index_file A text-file that determines the relationship between on the one hand the images in infile and on the other hand the acquisition parameters in acq_file.
bvecs A text file with normalised vectors describing the direction of the diffusion weighting.
bvals A text file with b-values describing the "amount of" diffusion weighting

topup This should only be specified if you have previously run ‘topup’ on your data and should be the same name that you gave as an argument to the –out parameter when you ran topup, aka the base name for output files from topup.

outfile Output file basename
retimg (logical) return image of class nifti
opts Additional options to pass to arguments passed to eddy
verbose print diagnostic messages
eddy_cmd The version of eddy to run.
... Not currently used

Value

Result from system command currently
**eddy_correct**  
*Eddy Current Correction*

**Description**

This function calls `eddy_correct` from FSL for DTI Processing

**Usage**

```r
eddy_correct(infile, outfile = NULL, retimg = TRUE, reference_no = 0, ...)
```

**Arguments**

- `infile` input filename of 4D image.
- `outfile` Output filename
- `retimg` (logical) return image of class nifti
- `reference_no` Set the volume number for the reference volume that will be used as a target to register all other volumes to. (default=0, i.e. the first volume)
- `...` Additional arguments passed to `fslcmd`

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

---

**enforce_form**  
*Enforce Either Qform or Sform is set*

**Description**

Enforce Either Qform or Sform is set

**Usage**

```r
enforce_form(file, ...)
```

**Arguments**

- `file` (character) image filename or character of class nifti
- `...` additional arguments to pass to `getForms`

**Value**

A character filename
Examples

```r
if (have_fsl()) {
    res = enforce_form(mni_fname())
}
```

---

**face_removal_mask**  
*Face Removal Mask*

**Description**

Face Removal Mask

**Usage**

```r
deface_image(file, ...)  
```

**Arguments**

- **file**: input image
- **template**: Template image to register input image to. Set to NULL (recommended) if want to use from [https://github.com/poldracklab/pydeface](https://github.com/poldracklab/pydeface). Alternatively, use `mni_fname`.
- **face_mask**: Mask of image, in same space as `template`. Set to NULL (recommended) if want to use from [https://github.com/poldracklab/pydeface](https://github.com/poldracklab/pydeface). Alternatively, use `mni_face_fname`.
- **outfile**: Output file name
- **dof**: (numeric) degrees of freedom (default 6 - rigid body)
- **cost**: Cost function passed to flirt
- **retimg**: (logical) return image of class nifti
- **...**: not used

**Value**

An image or filename depending on `retimg`
Examples

```r
if (have_fsl()) {
  file = "~/Downloads/sample_T1_input.nii.gz"
  if (file.exists(file)) {
    mask = face_removal_mask(file = file,
      template = NULL, face_mask = NULL)
    image = fslmask(file, mask)
  }
}
```

---

**fast FSL FAST**

### Description

This function calls `fast` from FSL

### Usage

```r
fast(
  file,
  outfile = NULL,
  bias_correct = TRUE,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  type = c("T1", "T2", "PD"),
  out_type = c("seg", "mixeltype", "pve_0", "pve_1", "pve_2", "pveseg"),
  verbose = TRUE,
  all_images = FALSE,
  ...
)
fast_all(..., all_images = TRUE)
fast_nobias_all(..., bias_correct = FALSE, all_images = FALSE)
fsl_fast(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
fslfast(...)
fsl_fast_nobias(
  ...,
  bias_correct = FALSE,
```
fast.help

```
ofile = tempfile(fileext = ".nii.gz"),
  retimg = FALSE
)

fast_nobias(..., bias_correct = FALSE)

fslfast_nobias(..., bias_correct = FALSE)
```

Arguments

- `file` (character) image to be manipulated
- `outfile` (character) resultant image name (optional)
- `bias_correct` (logical) if FALSE, then "--nobias" is passed to FAST. Additional options can be sent using opts, but this is the most commonly one changed.
- `retimg` (logical) return image of class nifti
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical) to be passed to `system`
- `opts` (character) operations to be passed to `fast`
- `type` (character) type of image T1, T2, or PD.
- `out_type` (character) Suffix to grab from outfile. For example, output filename is `paste0(outfile, ", out_type)`
- `verbose` (logical) print out command before running
- `all_images` If retimg
- `...` additional arguments passed to `readnii`

Value

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

```

Description

This function calls fast’s help

Usage

`fast.help()`
Value
  Prints help output and returns output as character vector

Examples
  if (have.fsl()){
    fast.help()
  }

flirt  Register using FLIRT

Description
  This function calls flirt to register infile to reffile and either saves the image or returns an object
  of class nifti, along with the transformation matrix omat

Usage
  flirt(
    infile, reffile,
    omat = NULL,
    dof = 6,
    outfile = NULL,
    retimg = TRUE,
    reorient = FALSE,
    intern = FALSE,
    opts = "",
    verbose = TRUE,
    ...
  )

Arguments
  infile  (character) input filename
  reffile  (character) reference image to be registered to
  omat    (character) Output matrix name
  dof      (numeric) degrees of freedom (default 6 - rigid body)
  outfile (character) output filename
  retimg   (logical) return image of class nifti
  reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
  intern   (logical) pass to system
  opts     (character) additional options to FLIRT
  verbose (logical) print out command before running
  ...      additional arguments passed to readnii.
**flirt.help**

**Value**

character or logical depending on intern

**Description**

This function calls flirt's help

**Usage**

flirt.help()

**Value**

Prints help output and returns output as character vector

**Examples**

```r
if (have.fsl()){  
  flirt.help()  
}
```

**flirt_apply**

*Apply Warp from FLIRT*

**Description**

This function applies a matrix from flirt to other images

**Usage**

```r
flirt_apply(  
  infile,  
  reffile,  
  initmat,  
  outfile = NULL,  
  retimg = TRUE,  
  reorient = FALSE,  
  intern = FALSE,  
  opts = "",  
  verbose = TRUE,  
  ...  
)
```
Arguments

infile (character) input filename
reffile (character) reference image to be registered to
initmat (character) Matrix of transformation
outfile (character) output filename
reting (logical) return image of class nifti
reorient (logical) If reting, should file be reoriented when read in? Passed to readnii.
intern (logical) pass to system
opts (character) additional options to FLIRT
verbose (logical) print out command before running
...
additional arguments passed to readnii.

Value

character or logical depending on intern

fnirt

Register using FNIRT

Description

This function calls fnirt to register infile to reffile and either saves the image or returns an object of class nifti

Usage

fnirt(
  infile,
  reffile,
  outfile = NULL,
  reting = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)
)
fnirt.help

Arguments

- `infile` (character) input filename
- `reffile` (character) reference image to be registered to
- `outfile` (character) output filename
- `reorient` (logical) return image of class nifti
- `intern` (logical) If reorient, should file be reoriented when read in? Passed to `readnii`.
- `opts` (character) additional options to FLIRT
- `verbose` (logical) print out command before running
- `...` additional arguments passed to `readnii`.

Value

character or logical depending on intern

fnirt.help

FNIRT help

Description

This function calls fnirt’s help

Usage

fnirt.help()

Value

Prints help output and returns output as character vector

fnirt_with_affine

Register using FNIRT, but doing Affine Registration as well

Description

This function calls fnirt to register infile to reffile and either saves the image or returns an object of class nifti, but does the affine registration first
Usage

fnirt_with_affine(
  infile,
  reffile,
  flirt.omat = NULL,
  flirt.outfile = NULL,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  flirt.opts = "",
 opts = "",
  verbose = TRUE,
  ...
)

Arguments

infile         (character) input filename
reffile         (character) reference image to be registered to
flirt.omat      (character) Filename of output affine matrix
flirt.outfile   (character) Filename of output affine-registered image
outfile         (character) output filename
retimg          (logical) return image of class nifti
reorient        (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern          (logical) pass to system
flirt.opts      (character) additional options to FLIRT
opts            (character) additional options to FNIRT
verbose         (logical) print out command before running
...              additional arguments passed to readnii.

Value

character or logical depending on intern

fnirt_with_affine_apply

Applies FLIRT then FNIRT transformations

Description

Applies an affine transformation with FLIRT then the warp image with FNIRT
Usage

fnirt_with_affine_apply(
  infile,
  reffile,
  flirt.omat = NULL,
  flirt.outfile = NULL,
  fnirt.warpfile = NULL,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  flirt.opts = "",
  opts = "",
  verbose = TRUE,
  ...
)

Arguments

infile       (character) input filename
reffile      (character) reference image to be registered to
flirt.omat   (character) Filename of output affine matrix
flirt.outfile (character) Filename of output affine-registered image
fnirt.warpfile (character) Filename of warp image from fnirt
outfile      (character) output filename
retimg       (logical) return image of class nifti
reorient     (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern       (logical) pass to system
flirt.opts   (character) additional options to FLIRT
opts         (character) additional options to FNIRT
verbose      (logical) print out command before running
...           additional arguments passed to readnii.

Value

character or logical depending on intern

See Also

fnirt_with_affine
Description

This function calls fslmaths's help, as fslabs is a wrapper for fslmaths

Usage

fslabs.help(...)  

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fslabs.help()
}

fslacos.help  

Description

This function calls fslmaths's help, as fslacos is a wrapper for fslmaths

Usage

fslacos.help(...)  

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fslacos.help()
}
**fsladd.help**

**Description**

This function calls `fslmaths`'s help, as `fsladd` is a wrapper for `fslmaths`.

**Usage**

```r
fsladd.help(...) 
```

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector.

**Examples**

```r
if (have.fsl()){
  fsladd.help()
}
```

---

**fsland**

*Logical AND with Images using FSL*

**Description**

This function multiplies two images using `fslmul` after binarizing the images (using `fslbin`).

**Usage**

```r
fsland(file, file2, ...) 
```

```r
fsl_and(file, file2, ...) 
```

**Arguments**

- `file` (character) input image
- `file2` (character) image to be multiplied
- `...` additional arguments passed to `fslmul`. 

fslasin.help

Value
If `retimg` then object of class nifti. Otherwise, result from system command, depends if `intern` is TRUE or FALSE.

Note
Functions with underscores have different defaults and will return an output filename, so to be used for piping

 Usage
fslasin.help(...)  

Arguments
... passed to `fslmaths.help`

Value
Prints help output and returns output as character vector

Examples
if (have.fsl()){
  fslasin.help()
}

fslatan.help

Description
This function calls fslmaths's help, as fslasin is a wrapper for fslmaths

Usage
fslatan.help(...)

Description
This function calls fslmaths's help, as fslatan is a wrapper for fslmaths

Usage
fslatan.help(...)


Arguments

... passed to \texttt{fslmaths}\texttt{.help}

Value

Prints help output and returns output as character vector

Examples

\begin{verbatim}
if (have.fsl()){
  fslatan.help()
}
\end{verbatim}
Description

This function calls fslmaths's help, as fslbin is a wrapper for fslmaths

Usage

fslbin.help(…)

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){  
  fslbin.help()  
}

Description

This function calls fslmaths's help, as fslbinv is a wrapper for fslmaths

Usage

fslbinv.help(…)

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){  
  fslbinv.help()  
}
Description

This function calls `fslchfiletype`

Usage

```r
fslchfiletype(
  file,
  filetype = "NIFTI_GZ",
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

- `file` (character) image to be manipulated
- `filetype` (character) filetype to change image to
- `outfile` (character) Output filename. If NULL, will overwrite input file
- `retimg` (logical) return image of class nifti
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical) to be passed to `system`
- `verbose` (logical) print out command before running
- `...` additional arguments passed to `readnii`

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.
fslchfiletype.help  

Description

This function calls fslchfiletype’s help

Usage

fslchfiletype.help()

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fslchfiletype.help()
}

fslcmd  

FSL Command Wrapper

Description

This function calls fsl command passed to func

Usage

fslcmd(
  func,
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  samefile = FALSE,
  opts_after_outfile = FALSE,
  frontopts = "",
  no.outfile = FALSE,
  trim_front = FALSE,
  run = TRUE,
  ...
)

Arguments

- `func` (character) FSL function
- `file` (character) image to be manipulated
- `outfile` (character) resultant image name (optional)
- `reorient` (logical) return image of class nifti
- `intern` (logical) If reorient, should file be reoriented when read in? Passed to `readnii`.
- `opts` (character) operations to be passed to `func`
- `verbose` (logical) print out command before running
- `samefile` (logical) is the output the same file?
- `opts_after_outfile` (logical) should opts come after the outfile in the FSL command?
- `frontopts` (character) options/character to put in before filename
- `nooutfile` (logical) is there an output file in the arguments of the FSL function?
- `trim_front` trim the whitespace from the front of the command.
- `run` (logical) Should the command just be printed (if FALSE)?
- ... additional arguments passed to `readnii`.

Value

If `reorient` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

---

**fslcog**

*Image Center of Gravity (FSL)*

Description

Find Center of Gravity of Image from FSL

Usage

`fslcog(img, mm = TRUE, verbose = TRUE, ts = FALSE)`

Arguments

- `img` Object of class nifti, or path of file
- `mm` Logical if the center of gravity (COG) would be in mm (default TRUE) or voxels (FALSE)
- `verbose` (logical) print out command before running
- `ts` (logical) is the series a timeseries (4D), invoking -t option
fslcos.help

Value

Vector of length 3 unless ts option invoked

Note

FSL uses a 0-based indexing system, which will give you a different answer compared to cog, but fslcog(img, mm = FALSE) + 1 should be relatively close to cog(img)

Examples

if (have.fsl()){
  x = array(rnorm(1e6), dim = c(100, 100, 100))
  img = nifti(x, dim= c(100, 100, 100), 
         datatype = convert.datatype()$FLOAT32, cal.min = min(x),
         cal.max = max(x), pixdim = rep(1, 4))
  fslcog(img)
}

fslcos.help  

Description

This function calls fslmaths's help, as fslcos is a wrapper for fslmaths

Usage

fslcos.help(...)

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fslcos.help()
}
**fslcpgeom**  

**FSL Copy Geometry**

**Description**

This function calls `fslcpgeom`

**Usage**

```r
fslcpgeom(
  file,
  file_with_header,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)
```

**Arguments**

- `file` (character) image to be manipulated
- `file_with_header` image with header to be copied over
- `retimg` (logical) return image of class nifti
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- `intern` (logical) to be passed to `system`
- `opts` (character) operations to be passed to `fslmaths`
- `verbose` (logical) print out command before running
- `...` additional arguments passed to `readnii`.

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.
Description

This function calls fslcpgeom’s help

Usage

fslcpgeom.help()

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fslcpgeom.help()
}

fsldir

Get FSL’s Directory

Description

Finds the FSLDIR from system environment or getOption("fsl.path") for location of FSL functions and returns it

Usage

fsldir()

fsl_dir()

Value

Character path


fsldiv.help

Description

This function calls fslmaths's help, as fsldiv is a wrapper for fslmaths.

Usage

fsldiv.help(...)

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector.

Examples

if (have.fsl()){
  fsldiv.help()
}

fsledge.help

Description

This function calls fslmaths's help, as fsledge is a wrapper for fslmaths.

Usage

fsledge.help(...)

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector.

Examples

if (have.fsl()){
  fsledge.help()
}
### fslentropy

**Description**

Estimates Mean Entropy of Image from FSL.

**Usage**

```r
fslentropy(img, nonzero = FALSE, verbose = TRUE, ts = FALSE)
```

**Arguments**

- `img` Object of class nifti, or path of file
- `nonzero` (logical) Should the statistic be taken over non-zero voxels
- `verbose` (logical) print out command before running
- `ts` (logical) is the series a timeseries (4D), invoking -t option

**Value**

Vector of unless ts option invoked, then matrix

**Note**

This uses option -e or -E in `fslstats`

### fslepi_reg

**Description**

This function calls epi_reg, designed to register EPI images (typically functional or diffusion) to structural (e.g. T1-weighted) image.

**Usage**

```r
fslepi_reg(  
  epi,  
  t1,  
  t1_brain,  
  outfile = NULL,  
  retimg = TRUE,  
  reorient = FALSE,  
  intern = FALSE,  
  fmap = NULL,  
)```
fmap_mag = NULL,
fmap_mag_brain = NULL,
echo_spacing = NA,
phase_enc_dir = c("x", "y", "z", "-x", "-y", "-z"),
weight = NULL,
verbose = TRUE,
opts = ",",
...
)

fsl_epi_reg(., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

epi_reg(., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

Arguments

epi EPI image, character or nifti object
t1 whole head T1 image , character or nifti object
t1_brain brain extracted T1 image
outfile output registered image filename
retimg (logical) return image of class nifti
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
fmap fieldmap image (in rad/s)
fmap_mag fieldmap magnitude image - whole head extracted
fmap_mag_brain fieldmap magnitude image - brain extracted
echo_spacing Effective EPI echo spacing (sometimes called dwell time) - in seconds
phase_enc_dir phase encoding direction, dir = x/y/z/-x/-y/-z
weight weighting image (in T1 space)
verbose (logical) print out command before running
opts (character) operations to be passed to fslmaths
... additional arguments passed to readnii.

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping.
fslerode.help  

**Description**

This function calls `fslmaths`'s help, as `fslerode` is a wrapper for `fslmaths`.

**Usage**

```r
fslerode.help(...)  
```

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector.

**Examples**

```r
if (have.fsl()){
  fslerode.help()
}
```

fslexp.help  

**Description**

This function calls `fslmaths`'s help, as `fslexp` is a wrapper for `fslmaths`.

**Usage**

```r
fslexp.help(...)  
```

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector.

**Examples**

```r
if (have.fsl()){
  fslexp.help()
}
```
Description

This function calls fslmaths's help, as fslfill is a wrapper for fslmaths.

Usage

fslfill.help(...)

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
    fslfill.help()
    fslfill2()
}

Description

This function calls fslmaths to dilate an image, then calls it again to erode it.

Usage

fslfill2(
    file,
    outfile = NULL,
    kopts = "",
    remove.ends = TRUE,
    refill = TRUE,
    reorient = TRUE,
    intern = FALSE,
    verbose = TRUE,
    ...
)
fslgetorient

Arguments

- `file` (character) filename of image to be filled
- `outfile` (character) name of resultant filled file
- `kopts` (character) Options passed for kernel before erosion/dilation
- `remove.ends` (logical) Remove top and bottom dilation.
- `refill` (logical) Run `fslfill` after dilation/erosion.
- `reorient` (logical) Return image of class nifti
- `reorient` (logical) If reorienting, should file be reoriented when read in? Passed to `readnii`.
- `intern` (logical) pass to `system`
- `verbose` (logical) print out command before running
- `...` additional arguments passed to `readnii`.

Value

Character or logical depending on `intern`

Note

This function binarizes the image before running.

fslgetorient FSL Orientation Wrappers

Description

This function calls `fslorient -get*` and is a simple wrapper of `fslorient`.

Usage

- `fslgetorient(file, verbose = TRUE)`
- `fslgetsform(file, verbose = TRUE)`
- `fslgetqform(file, verbose = TRUE)`
- `fslgetsformcode(file, verbose = TRUE)`
- `fslgetqformcode(file, verbose = TRUE)`

Arguments

- `file` (character) image to be manipulated
- `verbose` (logical) print out command before running

Value

Result from system command, output from FSL.
fslhd  

Get NIfTI header using FSL

Description

This function calls fslhd to obtain a nifti header

Usage

fslhd(file, opts = "", verbose = TRUE, ...)

Arguments

  file (character) image filename or character of class nifti
  opts (character) additional options to be passed to fslhd
  verbose (logical) print out command before running
  ... options passed to checkimg

Value

Character of information from fslhd

Examples

if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard",
    "MN1152_T1_2mm.nii.gz")
  fslhd(mnifile)
}

fslhd.help  

FSLhd help

Description

This function calls fslhd’s help

Usage

fslhd.help()

Value

Prints help output and returns output as character vector
Examples

if (have.fsl()){
  fslhd.help()
}

---

**fslhd.parse**  
*Parse FSL Header*

**Description**

This function takes in a FSL header and parses the components.

**Usage**

fslhd.parse(hd)

**Arguments**

hd  
(character) header from *fslhd*

**Value**

data.frame of information from FSL header

**Examples**

if (have.fsl()){
  mni_file = mni_fname("2")
  hd = fslhd(mni_file)
  fslhd.parse(hd)
}

---

**fslhelp**  
*Wrapper for getting fsl help*

**Description**

This function takes in the function and returns the help from FSL for that function.

**Usage**

fslhelp(func_name, help.arg = "--help", extra.args = "")
Arguments

- **func_name**: FSL function name
- **help.arg**: Argument to print help, usually "--help"
- **extra.args**: Extra arguments to be passed other than `--help`

Value

Prints help output and returns output as character vector

---

**fslindex.help**

**fslindex Help**

Description

This function calls `fslmaths`'s help, as `fslindex` is a wrapper for `fslmaths`

Usage

`fslindex.help(...)`

Arguments

... passed to `fslmaths.help`

Value

Prints help output and returns output as character vector

Examples

```r
if (have.fsl()){  
  fslindex.help()  
}
```
fsllog.help

Description
This function calls fslmaths’s help, as fsllog is a wrapper for fslmaths

Usage
fsllog.help(...)

Arguments
... passed to fslmaths.help

Value
Prints help output and returns output as character vector

Examples
if (have.fsl()){
  fsllog.help()
}

fslmask.help

Description
This function calls fslmaths’s help, as fslmask is a wrapper for fslmaths

Usage
fslmask.help(...)

Arguments
... passed to fslmaths.help

Value
Prints help output and returns output as character vector

Examples
if (have.fsl()){
  fslmask.help()
}
Description

This function calls fslmaths’s help

Usage

fslmaths.help()

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fslmaths.help()
}

desc

Description

This function calls the range or robust range functions from FSL and then extracts the min/max

Usage

fslmax(file, ...)
fslmin(file, ...)

Arguments

file (character) filename of image to be checked
... options passed to fslrange

Value

Numeric vector of mins/maxs or just one depending if ts = TRUE
Examples

```r
if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard", "MNI152_T1_2mm.nii.gz")
  fslmax(mnifile)
}
```

---

**fslmean**

*Image Mean*

---

**Description**

Estimates Mean of Image from FSL

**Usage**

`fslmean(img, nonzero = FALSE, verbose = TRUE, ts = FALSE)`

**Arguments**

- `img` Object of class nifti, or path of file
- `nonzero` (logical) Should the statistic be taken over non-zero voxels
- `verbose` (logical) print out command before running
- `ts` (logical) is the series a timeseries (4D), invoking `-t` option

**Value**

Vector of unless ts option invoked, then matrix

**Note**

This uses option -m or -M in `fslstats`

---

**fslmerge.help**

*FSLMerge help*

---

**Description**

This function calls fslmerge's help

**Usage**

`fslmerge.help()`
\section*{fslmul.help}

\section*{Value}
Prints help output and returns output as character vector

\section*{Examples}
\begin{verbatim}
if (have.fsl()){
  fslmerge.help()
}
\end{verbatim}

\section*{fslnan.help}

\section*{Description}
This function calls fslmaths's help, as fslnan is a wrapper for fslmaths

\section*{Usage}
\begin{verbatim}
fslnan.help(...)\end{verbatim}

\section*{Arguments}
\begin{verbatim}
... passed to fslmaths.help\end{verbatim}

\section*{Value}
Prints help output and returns output as character vector

\section*{Examples}
\begin{verbatim}
if (have.fsl()){
  fslnan.help()
}
\end{verbatim}
Arguments

... passed to \texttt{fslmaths.help}

Value

Prints help output and returns output as character vector

Examples

\begin{verbatim}
if (have.fsl()){
  fsl NaNm.help()
}\end{verbatim}
fslor

Perform OR/Union operation on Images using FSL

Description

This function calls fslmaths file -add file2 -bin after binarizing file and file2 using fslbin.

Usage

```r
fslor(
  file,
  file2,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  ...
)
```

```r
fsl_or(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

Arguments

- **file** (character) input image
- **file2** (character) image to be unioned
- **outfile** (character) resultant image name (optional)
- **retimg** (logical) return image of class nifti
- **reorient** (logical) If retimg, should file be reoriented when read in? Passed to readnii.
- **intern** (logical) to be passed to system
- **...** additional arguments passed to readnii.

Value

- If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping.
fslorient  

FSL Orient

Description
This function calls fslorient

Usage
fslorient(
  file,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = '',
  verbose = TRUE,
  ...
)

Arguments
  file (character) image to be manipulated
  retimg (logical) return image of class nifti
  reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
  intern (logical) to be passed to system
  opts (character) operations to be passed to fslorient
  verbose (logical) print out command before running
  ... additional arguments passed to readnii.

Value
If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

---

fslorient.help  

fslorient help

Description
This function calls fslorient's help

Usage
fslorient.help()
Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){  
  fslorient.help()  
}

fslorienter

Wrapper for FSL Get Orientation

Description

This function calls fslorient -getorient and is a simple wrapper of fslorient

Usage

fslorienter(file, opts = "", verbose = TRUE)

Arguments

file (character) image to be manipulated
opts option to send to fslorient
verbose (logical) print out command before running

Value

Result from system command, output from FSL

fslrand.help

fslrand Help

Description

This function calls fslmaths's help, as fslrand is a wrapper for fslmaths

Usage

fslrand.help(...) 

Arguments

... passed to fslmaths.help
Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
    fslrand.help()
}

---

fslrandn.help fslrandn Help

Description

This function calls fslmaths’s help, as fslrandn is a wrapper for fslmaths

Usage

fslrandn.help(...)

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
    fslrandn.help()
}

---

fslrange Get range of an image

Description

This function calls fslstats -R to get the range of an image or fslstats -r to get the robust range

Usage

fslrange(file, robust = FALSE, verbose = TRUE, ts = FALSE, ...)

---
Arguments

- **file**: (character) filename of image to be checked
- **robust**: (logical) Should the range be robust (-r)
- **verbose**: (logical) print out command before running
- **ts**: (logical) is the series a timeseries (4D), invoking -t option
- **...**: options passed to `checkimg`

Value

numeric vector of length 2

Examples

```r
if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard",
                      "MNI152_T1_2mm.nii.gz")
  fslrange(mnifile)
}
```

---

**Description**

This function calls `fslmaths`'s help, as `fslrecip` is a wrapper for `fslmaths`

**Usage**

`fslrecip.help()`

**Arguments**

- **...**: passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector

**Examples**

```r
if (have.fsl()){
  fslrecip.help()
}
```
Description
This function calls fslmaths’s help, as fslrem is a wrapper for fslmaths

Usage
fslrem.help(...)  

Arguments
... passed to fslmaths.help  

Value
Prints help output and returns output as character vector

Examples
if (have.fsl()){
  fslrem.help()
}  

Description
This function calls fslreorient2std

Usage
fslreorient2std(
  file,  
  retimg = TRUE,  
  reorient = FALSE,  
  intern = FALSE,  
  verbose = TRUE,  
  opts = "",  
  ...
)

fslreorient2std_mat(
  file,  
}
Arguments

file (character) image to be manipulated
retimg (logical) return image of class nifti
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
verbose (logical) print out command before running
opts additional options to pass to fslreorient2std
... additional arguments passed to readnii.
matfile Output file for the matrix for reorientation

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Description

This function calls fslreorient2std's help

Usage

fslreorient2std.help()

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fslreorient2std.help()
}
fslrobustfov  FSL Robust Field of View

Description

This function calls robustfov to automatically crop the image

Usage

fslrobustfov(
  file,
  brain_size = NULL,
  mat_name = NULL,
  roi_name = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  verbose = TRUE,
  ...
)

fsl_robustfov(retimg = FALSE, ...)

Arguments

file  (character) image to be manipulated
brain_size  size of brain in z-dimension (default 150mm)
mat_name  matrix output name
roi_name  ROI volume output name
retimg  (logical) return image of class nifti
reorient  (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern  (logical) to be passed to system
verbose  (logical) print out command before running
...  additional arguments passed to readnii.

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.
### Description

This function calls `robustfov help`

### Usage

```r
fslrobustfov.help()
```

### Description

This function calls `fslroi`

### Usage

```r
fslroi(
    file,
    xmin = 0,
    xsize = -1,
    ymin = 0,
    ysize = -1,
    zmin = 0,
    zsize = -1,
    tmin = NULL,
    tsize = NULL,
    outfile = NULL,
    retimg = TRUE,
    reorient = FALSE,
    intern = FALSE,
    verbose = TRUE,
    ...
)
```

```r
fsl_roi(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fslroi_time(file, tmin = NULL, tsize = NULL, ...)
```
**Arguments**

- **file** (character) image to be manipulated
- **xmin** Minimum index for x-dimension
- **xsize** Size of ROI in x-dimension
- **ymin** Minimum index for y-dimension
- **ysize** Size of ROI in y-dimension
- **zmin** Minimum index for z-dimension
- **zsize** Size of ROI in z-dimension
- **tmin** Minimum index for t-dimension
- **tsize** Size of ROI in t-dimension
- **outfile** (character) resultant image name (optional)
- **retimg** (logical) return image of class nifti
- **reorient** (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- **intern** (logical) to be passed to `system`
- **verbose** (logical) print out command before running
- **...** additional arguments passed to `readnii`.

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

**Note**

Indexing (in both time and space) starts with 0 not 1! Inputting -1 for a size will set it to the full image extent for that dimension.

---

**fslsd**

*Image Standard Deviation*

**Description**

Estimates Standard Deviation of Image from FSL

**Usage**

```r
fslsd(img, nonzero = FALSE, verbose = TRUE, ts = FALSE)
```

**Arguments**

- **img** Object of class nifti, or path of file
- **nonzero** (logical) Should the statistic be taken over non-zero voxels
- **verbose** (logical) print out command before running
- **ts** (logical) is the series a timeseries (4D), invoking -t option
fslsin

Value

Vector of unless ts option invoked, then matrix

Note

This uses option -s or -S in fslstats

fslsin

Sine Transform Image using FSL

Description

This function calls fslmaths -sin. The R functions wraps fslmaths

Usage

fslsin(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments

file (character) input image to sine transform
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths
... additional arguments passed to readnii.

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.
fslsin.help  
fslsin Help

Description
This function calls fslmaths's help, as fslsin is a wrapper for fslmaths

Usage
fslsin.help(...)

Arguments
... passed to fslmaths.help

Value
Prints help output and returns output as character vector

Examples
if (have.fsl()){
  fslsin.help()
}

fslslicetimer  
FSL Slice Timing Correction

Description
This function calls slicetimer and performs slice timing correction for fMRI data

Usage
fslslicetimer(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  tr = 3,
  direction = "z",
  indexing = c("up", "down"),
  acq_order = c("contiguous", "interleaved"),
  verbose = TRUE,
...
Arguments

- **file**: (character) image to be manipulated
- **outfile**: (character) resultant image name (optional)
- **reorient**: (logical) return image of class nifti
- **reorient**: (logical) If reorient, should file be reoriented when read in? Passed to `readnii`.
- **intern**: (logical) to be passed to `system`
- **tr**: (numeric) Repeat time in seconds
- **direction**: (character) Direction of acquisition
- **indexing**: (character) Whether indexing was bottom up (default) or down using `--down` option
- **acq_order**: (character) Order of acquisition, either contiguous or interleaved
- **verbose**: (logical) print out command before running
- **...**: additional arguments passed to `readnii`.

Value

If `reorient` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

**fslsmooth.help**  
**fslsmooth Help**

**Description**

This function calls fslmaths's help, as fslsmooth is a wrapper for fslmaths

**Usage**

`fslsmooth.help(...)`

**Arguments**

- `...` passed to `fslmaths.help`
Value

Prints help output and returns output as character vector

Examples

```r
if (have.fsl()){  
  fslsmooth.help()  
}
```

---

**fslsmooth_in_mask**  
*Smooth Image Within a Mask Only*

---

Description

This function smooth an image within a mask and replaces the values of the original image with the smoothed values.

Usage

```r
fslsmooth_in_mask(file, sigma = 10, mask = NULL, ...)
```

```r
fsl_smooth_in_mask(...)```

Arguments

- `file` (character) image to be smoothed
- `sigma` (numeric) sigma (in mm) of Gaussian kernel for smoothing
- `mask` (character) optional mask given for image
- `...` additional arguments passed to `fslsmooth`.

Value

Object of class `nifti`

Examples

```r
if (have.fsl()){  
  system.time({  
    dims = c(50, 50, 20)  
    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))  
    mask = abs(img) > 1  
    s.img = fslsmooth_in_mask(img, mask = mask)  
  })  
}
Split images using FSL

Description

This function calls fslsplit to merge files on some dimension and either saves the image or returns an object of class nifti

Usage

fslsplit(
  infile,
  direction = c("t", "x", "y", "z"),
  output basename = NULL,
  retimg = TRUE,
  reorient = FALSE,
  verbose = TRUE
)

fsl_split(..., retimg = FALSE)

Arguments

infile (character) input filename
direction (character) direction to split over: t (time), x, y, z
output basename (character) prefix to have for output
retimg (logical) return image of class nifti
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
verbose (logical) print out command before running
... not used

Value

List of output files

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping
fslsplit.help  

**FSL Split help**

**Description**
This function calls fslsplit’s help

**Usage**
```r
fslsplit.help()
```

**Value**
Prints help output and returns output as character vector

**Examples**
```r
if (have.fsl()){  
    fslsplit.help()
}
```

---

fslsqr.help  

**fslsqr Help**

**Description**
This function calls fslmaths’s help, as fslsqr is a wrapper for fslmaths

**Usage**
```r
fslsqr.help(...)
```

**Arguments**

... passed to `fslmaths.help`

**Value**
Prints help output and returns output as character vector

**Examples**
```r
if (have.fsl()){  
    fslsqr.help()
}
```
Description

This function calls fslmaths's help, as fslsqrt is a wrapper for fslmaths.

Usage

fslsqrt.help(...)

Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector.

Examples

if (have.fsl()){
  fslsqrt.help()
}

fslstats

FSL Stats

Description

This function calls fslstats.

Usage

fslstats(file, opts = "", verbose = TRUE, ts = FALSE, ...)

Arguments

file (character) filename of image to be checked
opts (character) operation passed to fslstats
verbose (logical) print out command before running
(ts (logical) is the series a timeseries (4D), invoking -t option
... options passed to checking
Value

Result of fslstats command

Examples

```r
if (have.fsl()){
  system.time(
    x = array(rnorm(1e6), dim = c(100, 100, 100))
    img = nifti(x, dim= c(100, 100, 100),
     datatype = convert.datatype()$FLOAT32, cal.min = min(x),
     cal.max = max(x), pixdim = rep(1, 4))
    entropy = fslstats(img, opts="-E")
  )
}
```

---

**fslstats.help**

---

**FSL Stats Help**

**Description**

This function calls fslstats’s help

**Usage**

`fslstats.help()`

**Value**

Prints help output and returns output as character vector

**Examples**

```r
if (have.fsl()){
  fslstats.help()
}
```

---

**fslsub.help**

---

**fslsub Help**

**Description**

This function calls fslmaths’s help, as fslsub is a wrapper for fslmaths

**Usage**

`fslsub.help(...)`
Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fslsub2.help()
}
**fslsum**  
*FSL Sum*

**Description**  
This function calls `fslstats -M -V` to get product, aka the approximate sum.

**Usage**  
```r  
fslsum(file, opts = "", ts = FALSE, ...)  
```

**Arguments**  
- `file` (character) filename of image to be checked  
- `opts` Additional options to pass to `fslstats`  
- `ts` (logical) is the series a timeseries (4D), invoking `-t` option  
- `...` options passed to `fslstats`

**Value**  
Numeric value

**Note**  
This may be approximate due to rounding

---

**fslswapdim.help**  
*fslswapdim help*

**Description**  
This function calls `fslswapdim`'s help

**Usage**  
```r  
fslswapdim.help()  
```

**Value**  
Prints help output and returns output as character vector

**Examples**  
```r  
if (have.fsl()){
  fslswapdim.help()
}
```
**fsltan.help**

**fsltan Help**

**Description**
This function calls fslmaths’s help, as fsltan is a wrapper for fslmaths

**Usage**
```
fsltan.help(...)```

**Arguments**
```
... passed to fslmaths.help```

**Value**
Prints help output and returns output as character vector

**Examples**
```
if (have.fsl()){  
  fsltan.help()  
}
```

---

**fslthresh.help**

**fslthresh Help**

**Description**
This function calls fslmaths’s help, as fslthresh is a wrapper for fslmaths

**Usage**
```
fslthresh.help(...)```

**Arguments**
```
... passed to fslmaths.help```

**Value**
Prints help output and returns output as character vector

**Examples**
```
if (have.fsl()){  
  fslthresh.help()  
}
```
**fslval**  
*Get value from FSL header*

**Description**
This function calls fslval to obtain a nifti header

**Usage**

```r
fslval(file, keyword = "", verbose = TRUE, ...)
```

**Arguments**
- `file` (character) image filename or character of class nifti
- `keyword` (character) keyword to be taken from fslhd
- `verbose` (logical) print out command before running
- `...` options passed to `checkimg`

**Value**
Character of information from fslhd field specified in keyword

**Examples**

```r
if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard",
                    "MNI152_T1_2mm.nii.gz"
  fslval(mnifile, keyword = "dim1")
}
```

---

**fslval.help**  
*fslval help*

**Description**
This function calls fslval's help

**Usage**

```r
fslval.help()
```

**Value**
Prints help output and returns output as character vector
Examples
if (have.fsl()){
  fslval.help()
}

Description
This function calls fslview to view an image in the FSL viewer

Usage
fslview(file, intern = TRUE, opts = "", verbose = TRUE, ...)
fsleyes(file, intern = TRUE, opts = "", verbose = TRUE, ...)

Arguments
file (character) filename of image to be thresholded
intern (logical) pass to system
opts (character) options for FSLView
verbose (logical) print out command before running
... options passed to checkimg

Value
character or logical depending on intern

Note
As of FSL version 5.0.10, this is deprecated: https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/WhatsNew
fslview.help

**FSLView help**

**Description**

This function calls fslview's help

**Usage**

fslview.help()

**Value**

Prints help output and returns output as character vector

**Examples**

library(fslr)
if (have.fsl()){
  print(fsl_version())
  in_ci <- function() {
    nzchar(Sys.getenv("CI"))
  }
  if (!in_ci()) {
    fslview.help()
  }
}

fslvol

**FSL Volume in mL (or cubic centimeters)**

**Description**

This function wraps fslsum and voxdim

**Usage**

fslvol(file, ...)

**Arguments**

- file (character) filename of image to be checked
- ... options passed to fslsum

**Value**

Numeric value of volume in mL
Note

This may be approximate due to rounding

---

**fslvolume**

*Image Volume*

---

**Description**

Estimates Volume of Image from FSL

**Usage**

```
fslvolume(img, nonzero = FALSE, verbose = TRUE, ts = FALSE)
```

**Arguments**

- `img` Object of class nifti, or path of file
- `nonzero` (logical) Should the statistic be taken over non-zero voxels
- `verbose` (logical) print out command before running
- `ts` (logical) is the series a timeseries (4D), invoking `-t` option

**Value**

Vector of unless ts option invoked, then matrix

**Note**

This uses option `-v` or `-V` in **fslstats**

---

**fslxor**

*Perform XOR/Exclusive Or operation on Images using FSL*

---

**Description**

This function calls `fslmaths file -add file2 -bin` after binarizing `file` and `file2` using `fslbin` and then uses `fsl_thresh` to threshold any values greater than 1 back to zero.
Usage

```r
fslxor(
  file,
  file2,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
...
)
```

```r
fsl_xor(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

Arguments

- `file` (character): input image
- `file2` (character): image to be XOR’ed
- `outfile` (character): resultant image name (optional)
- `retimg` (logical): return image of class nifti
- `reorient` (logical): If retimg, should file be reoriented when read in? Passed to `readnii`.
- `intern` (logical): to be passed to `system`.
- `...` (additional arguments passed to `readnii`).

Value

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping.

---

**fsl_abs**

*Absolute Value Image using FSL*

Description

This function calls `fslmaths -abs`. The R functions wraps `fslmaths`
Usage

fsl_abs(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslabs(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) input image to absolute value
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

fsl_acos  Arc Cosine Transform Image using FSL

Description

This function calls fslmaths -acos. The R functions wraps fslmaths
Usage

fsl_acos(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslacos(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) input image to arc cosine transform
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

fsl_add

Add Images using FSL

Description

This function calls fslmaths -add. The R functions wraps fslmaths
**Usage**

```r
default_add(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

default_add(
  file,
  file2,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
```

**Arguments**

- `...` additional arguments passed to `readnii`.
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `file` (character) input image
- `file2` (character) image to be added
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- `intern` (logical) to be passed to `system`
- `opts` (character) operations to be passed to `fslmaths`

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

**Note**

Functions with underscores have different defaults and will return an output filename, so to be used for piping.

---

**fsl_anat**

*FSL Anatomical Processing Script*

**Description**

This function calls `fsl_anat` from FSL.
Usage

fsl_anat(
  file,
  modality = c("T1", "T2", "PD"),
  outdir = NULL,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)

Arguments

file (character) image to be manipulated, should be full path
modality (character) Modality of Image to be run
outdir (character) output directory, if none specified, will default to dirname(file)
intern (logical) to be passed to system
opts (character) operations to be passed to fsl_anat
verbose (logical) print out command before running
... options passed to checkimg

Value

Result from system command, depends if intern is TRUE or FALSE.

Description

This function calls fsl_anat’s help

Usage

fsl_anat.help()

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  fsl_anat.help()
}
Description

This function applies a coefficient map from fnirt to other images.

Usage

fsl_applywarp(
  infile,
  reffile,
  warpfile,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)

Arguments

infile (character) input filename
reffile (character) reference image to be registered to
warpfile (character) reference image to be registered to
outfile (character) output filename
retimg (logical) return image of class nifti
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) pass to system
opts (character) additional options to FLIRT
verbose (logical) print out command before running
... additional arguments passed to readnii.

Value

character or logical depending on intern
Description
This function calls applywarp’s help

Usage
fsl_applywarp.help()

Value
Prints help output and returns output as character vector

fsl_asin
Arc Sine Transform Image using FSL

Description
This function calls fslmaths -asin. The R functions wraps fslmaths

Usage
fsl_asin(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslasin(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments

...  additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) input image to arc sine transform
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths
Value

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

### fsl_atan

*Arc Tangent Transform Image using FSL*

#### Description

This function calls `fslmaths -atan`. The R functions wraps `fslmaths`

#### Usage

```r
fsl_atan(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fslatan(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
```

#### Arguments

- `...`: additional arguments passed to `readnii`.
- `outfile`: (character) resultant image name (optional)
- `retimg`: (logical) return image of class nifti
- `file`: (character) input image to arc tangent transform
- `reorient`: (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- `intern`: (logical) to be passed to `system`
- `opts`: (character) operations to be passed to `fslmaths`

#### Value

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.
Note
Functions with underscores have different defaults and will return an output filename, so to be used for piping.

---

**fsl_atlas_dir**  
*Get FSL’s Standard Data Directory*

**Description**
Finds the FSLDIR from system environment or getOption("fsl.path") and pastes on “data/standard”

**Usage**
```r
fsl_atlas_dir()
```

**Value**
Character path

---

**fsl_avscale**  
*Scale Affine Matrix using avscale*

**Description**
This function calls avscale to get individual matrices for FSL

**Usage**
```r
fsl_avscale(file, volume = NULL, parsed = TRUE, verbose = TRUE)
```

**Arguments**
- `file` (character) matrix filename
- `volume` (character) non-reference volume filename or nifti image
- `parsed` (logical) should `parse_avscale` be run after?
- `verbose` (logical) print out command before running
- `...` not used, but used for duplicating avscale as alias

**Value**
Character of information from avscale
Use FSL’s Brain Extraction Tool (BET)

Description

This function calls bet to extract a brain from an image, usually for skull stripping.

Usage

```r
fsl_bet(..., outfile = tempfile(fileext = "nii.gz"), retimg = FALSE)

fslbet(
  infile,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  betcmd = c("bet2", "bet"),
  verbose = TRUE,
  ...
)
```

Arguments

- `...` additional arguments passed to `readnii`
- `outfile` (character) output filename
- `retimg` (logical) return image of class `nifti`
- `infile` (character) input filename
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical) pass to `system`
- `opts` (character) additional options to bet
- `betcmd` (character) Use bet or bet2 function
- `verbose` (logical) print out command before running

Value

character or logical depending on `intern`

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping
**Description**

This function wraps a call to fast that performs bias correction.

**Usage**

```r
fsl_biascorrect(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  remove.seg = TRUE,
  ...
)
```

**Arguments**

- `file` (character) image to be manipulated
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical) to be passed to `system`
- `opts` (character) operations to be passed to `fast`
- `verbose` (logical) print out command before running
- `remove.seg` (logical) Should segmentation from FAST be removed?
- `...` additional arguments passed to `readnii`.

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.
Binarize Image using FSL

Description

This function calls fslmaths -bin. The R functions wraps fslmaths

Usage

fsl_bin(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslbin(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) image to be binarized
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping
Examples

```r
set.seed(5)
dims = rep(10, 3)
arr = array(rnorm(prod(dims)), dim = dims)
nim = oro.nifti::nifti(arr)
if (have.fsl()){
fslbin(nim)
fsl_bin(nim)
}
```

---

**fsl_binv**

*Binarized Inverse Image using FSL*

Description

This function calls *fslmaths* `-binv`. The R functions wraps *fslmaths*

Usage

```r
fsl_binv(..., outfile = tempfile(fileext = "nii.gz"), retimg = FALSE)
```

```r
fslbinv(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
```

Arguments

- `...`: additional arguments passed to `readnii`
- `outfile`: (character) resultant image name (optional)
- `retimg`: (logical) return image of class `nifti`
- `file`: (character) input image to take the binarized inverse
- `reorient`: (logical) If `retimg`, should file be reoriented when read in? Passed to `readnii`
- `intern`: (logical) to be passed to `system`
- `opts`: (character) operations to be passed to `fslmaths`

Value

If `retimg` then object of class `nifti`. Otherwise, Result from system command, depends if `intern` is `TRUE` or `FALSE`. 
**fsl_bin_tab**

**Quick Tabulation for logical images**

### Description

Creates a 2 by 2 table for

### Usage

```r
fsl_bin_tab(x, y, dnames = c("x", "y"), verbose = FALSE)
```

### Arguments

- **x**: filename of logical or 0/1 image
- **y**: filename of logical or 0/1 image
- **dnames**: names for table
- **verbose**: Should fsl commands be printed?

### Value

table of x vs y

### Note

fsl_bin will be run to make these images binary before running

---

**fsl_cluster**

Form clusters, report information about clusters and/or perform cluster-based inference. Wrapper for cluster

### Description

Form clusters, report information about clusters and/or perform cluster-based inference. Wrapper for cluster
Usage

```r
fsl_cluster(
  file,
  threshold,
  retimg = FALSE,
  reorient = FALSE,
  opts = "",
  cope_image = NULL,
  pthresh = NULL,
  peakdist = 0,
  volume = FALSE,
  smooth_est = NULL,
  voxel_resel = NULL,
  fractional = FALSE,
  connectivity = 26,
  mm = FALSE,
  find_minima = FALSE,
  standard_image = NULL,
  verbose = TRUE,
  ...
)
```

```r
fslcluster(..., retimg = TRUE)
```

```r
read_cluster_table(file)
```

Arguments

- **file**: filename of input volume
- **threshold**: threshold for input volume
- **retimg**: (logical) return image of class nifti
- **reorient**: (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- **opts**: (character) operations to be passed to `cluster`
- **cope_image**: filename of input cope volume
- **pthresh**: p-threshold
- **peakdist**: minimum distance between local maxima/minima, in mm (default 0)
- **volume**: number of voxels in the mask
- **smooth_est**: smoothness estimate = sqrt(det(Lambda))
- **voxel_resel**: Size of one resel in voxel units
- **fractional**: interprets the threshold as a fraction of the robust range
- **connectivity**: the connectivity of voxels (default 26)
- **mm**: use mm, not voxel, coordinates
- **find_minima**: find minima instead of maxima
- **standard_image**: filename for standard-space volume
verbose (logical) print out command before running
additional arguments to pass to fslcmd

Value
A list of filenames of outputs and tables:
  • opvals: filename for image output of log pvals
  • oindex: filename for output of cluster index (in size order)
  • othresh: filename for output of thresholded image
  • olmax: filename for output of local maxima text file
  • olmaxim: filename for output of local maxima volume
  • osize: filename for output of size image
  • omax: filename for output of max image
  • omean: filename for output of mean image

Examples
```r
if (have_fsl()) {
  file = mni_fname(brain = TRUE, mask = FALSE)
  threshold = 6000
  clus = fsl_cluster(file, threshold)
}
```

---

**fsl_cos**

Cosine Transform Image using FSL

Description
This function calls fslmaths \(-\text{cos}\). The R functions wraps fslmaths

Usage
```r
fsl_cos(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```
```r
fslcos(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
```
Arguments

... additional arguments passed to `readnii`.

outfile (character) resultant image name (optional)

reorient (logical) return image of class nifti

file (character) input image to cosine transform

reorient (logical) If reorient, should file be reoriented when read in? Passed to `readnii`.

intern (logical) to be passed to `system`

opts (character) operations to be passed to `fslmaths`

Value

If `reorient` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

`fsl_data_dir`  
*Get FSL’s Data Directory*

Description

Finds the FSLDIR from system environment or `getOption("fsl.path")` and pastes on “data”

Usage

`fsl_data_dir()`

Value

Character path
Tool to deface a structural T1w image.

**Usage**

```r
fsl_deface(
  file,
  outfile = NULL,
  retimg = TRUE,
  opts = "",
  deface_cropped = FALSE,
  bet_fractional_intensity = NULL,
  bias_correct = FALSE,
  shift_xyz = NULL,
  cog_xyz = NULL,
  reorient = FALSE,
  intern = FALSE,
  verbose = TRUE,
  ...
)
```

**Arguments**

- `file` (character) input image to estimate edge strength
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `opts` (character) operations to be passed to `fsl_deface`
- `deface_cropped` apply the defacing to the cropped image instead of the original image
- `bet_fractional_intensity` fractional intensity for bet (0->1); default=0.5;
- `bias_correct` Bias-correct the input image (with fast);
- `shift_xyz` Shift, in mm, x-, y- and z-directions, to shift face mask by;
- `cog_xyz` centre-of-gravity for bet (voxels, not mm);
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- `intern` (logical) to be passed to `system`
- `verbose` (logical) print diagnostic messages
- `...` additional arguments passed to `fslcmd`
Examples

```r
if (have_fsl()) {
  file = mni_fname(mm = 1, brain = FALSE)
  out = fsl_deface(file, retimg = FALSE)
}
```

---

**fsl_dice**

*Calculate Dice Coefficient of 2 Binary images*

**Description**

Creates a 2 by 2 table for

**Usage**

```r
fsl_dice(x, y, ...)
```

**Arguments**

- `x`: filename of logical or 0/1 image
- `y`: filename of logical or 0/1 image
- `...`: arguments passed to `fsl_bin_tab`

**Value**

Single number of the dice coefficient

---

**fsl_dilate**

*Dilate image using FSL*

**Description**

This function calls `fslmaths -ero` after inverting the image to dilate an image with either the default FSL kernel or the kernel specified in `kopts`. The function either saves the image or returns an object of class `nifti`. 
Usage

```r
fsl_dilate(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fsldilate(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  kopts = "",
  opts = "",
  verbose = TRUE,
  ...
)
```

Arguments

- `...`: additional arguments passed to `readnii`
- `outfile`: (character) resultant dilated image name
- `retimg`: (logical) return image of class nifti
- `file`: (character) image to be dilated
- `reorient`: (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- `intern`: (logical) to be passed to `system`
- `kopts`: (character) options for kernel
- `opts`: (character) additional options to be passed to `fslmaths`
- `verbose`: (logical) print out command before running

Value

Result from system command, depends if intern is TRUE or FALSE. If retimg is TRUE, then the image will be returned.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

Examples

```r
if (have.fsl()){
  system.time({
    dims = c(50, 50, 20)
    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))
    mask = img > .5
  })
```
dilated = fsldilate(mask, kopts = "-kernel boxv 5", retimg=TRUE)
}
}

---

**fsl_div**  
*Divide Images using FSL*

**Description**

This function calls *fslmaths* -div. The R functions wraps *fslmaths*

**Usage**

```r
fsl_div(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fsldiv(
    file,
    file2,
    outfile = NULL,
    retimg = TRUE,
    reorient = FALSE,
    intern = FALSE,
    opts = "",
    ...
)
```

**Arguments**

- `...` additional arguments passed to *readnii*.
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `file` (character) input image
- `file2` (character) image to be divided
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to *readnii*.
- `intern` (logical) to be passed to *system*
- `opts` (character) operations to be passed to *fslmaths*

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is `TRUE` or `FALSE`.

**Note**

Functions with underscores have different defaults and will return an output filename, so to be used for piping.
**fsl_edge**

*Edge Strength Image using FSL*

---

**Description**

This function calls `fslmaths -edge`. The R functions wraps `fslmaths`.

**Usage**

```r
fsl_edge(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fsledge(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
```

**Arguments**

- `...`: additional arguments passed to `readnii`.
- `outfile`: (character) resultant image name (optional)
- `retimg`: (logical) return image of class nifti
- `file`: (character) input image to estimate edge strength
- `reorient`: (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- `intern`: (logical) to be passed to `system`
- `opts`: (character) operations to be passed to `fslmaths`.

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is `TRUE` or `FALSE`.

**Note**

Functions with underscores have different defaults and will return an output filename, so to be used for piping.
fsl_erode  

Erode image using FSL

Description
This function calls fslmaths -ero to erode an image with either the default FSL kernel or the kernel specified in kopts. The function either saves the image or returns an object of class nifti.

Usage
fsl_erode(..., outfile = tempfile(fileext = "nii.gz"), retimg = FALSE)

fslerode(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  kopts = "",
  opts = "",
  verbose = TRUE,
  ...
)

Arguments

...  additional arguments passed to readnii.
outfile  (character) resultant eroded image name
retimg  (logical) return image of class nifti
file  (character) image to be eroded
reorient  (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern  (logical) to be passed to system
kopts  (character) options for kernel
opts  (character) additional options to be passed to fslmaths
verbose  (logical) print out command before running

Value
Result from system command, depends if intern is TRUE or FALSE. If retimg is TRUE, then the image will be returned.

Note
Functions with underscores have different defaults and will return an output filename, so to be used for piping
**Examples**

```r
if (have.fsl()){
  system.time({
    dims = c(50, 50, 20)
    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))
    mask = img > .5
    eroded = fslerode(mask, kopts = "-kernel boxv 5", retimg=TRUE)
  })
}
```

---

**fsl_exp**  
*Exponentiate Image using FSL*

---

**Description**

This function calls fslmaths -exp. The R functions wraps fslmaths

**Usage**

```r
fsl_exp(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fslexp(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
```

**Arguments**

- `...` additional arguments passed to `readnii`
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `file` (character) input image to exponentiated
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical) to be passed to `system`
- `opts` (character) operations to be passed to fslmaths
Value

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

**fsl_fill**  
*Fill image holes*

**Description**

This function calls `fslmaths -fillh` to fill in image holes and either saves the image or returns an object of class nifti

**Usage**

```r
fsl_fill(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fslfill(
  file,
  outfile = NULL,
  bin = TRUE,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  verbose = TRUE,
  ...
)
```

**Arguments**

- `...`: additional arguments passed to `readnii`
- `outfile` (character): name of resultant filled file
- `retimg` (logical): return image of class nifti
- `file` (character): filename of image to be filled
- `bin` (logical): binarize the image before filling
- `reorient` (logical): If `retimg`, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical): pass to `system`
- `verbose` (logical): print out command before running

**Value**

character or logical depending on `intern`
Note
Functions with underscores have different defaults and will return an output filename, so to be used for piping

Examples

```r
if (have.fsl()){
  system.time(
    dims = c(50, 50, 20)
    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))
    mask = img > .5
    eroded = fslerode(mask, kopts = "-kernel boxv 5", retimg=TRUE)
    filled = fslfill(eroded, retimg= TRUE)
  }
}
```

---

**fsl_index**

*Index Image using FSL*

**Description**

This function calls fslmaths -index. The R functions wraps fslmaths

**Usage**

```r
fsl_index(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fslindex(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
```

**Arguments**

- `...` additional arguments passed to `readnii`.
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `file` (character) input image to have non-zero entries replaced with index
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
fsl_log

Log Transform Image using FSL

Description

This function calls fslmaths -log. The R functions wraps fslmaths.

Usage

fsl_log(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fsllog(
    file,
    outfile = NULL,
    retimg = TRUE,
    reorient = FALSE,
    intern = FALSE,
    opts = "",
    ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) input image to log transform
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths
**Value**

If `retimg` then object of class `nifti`. Otherwise, Result from system command, depends if `intern` is `TRUE` or `FALSE`.

**Note**

Functions with underscores have different defaults and will return an output filename, so to be used for piping.

---

---

**Description**

This function calls `fslmaths -mas` to mask an image from an image mask and either saves the image or returns an object of class `nifti`.

**Usage**

```r
fsl_mask(..., outfile = tempfile(fileext = "nii.gz"), retimg = FALSE)
```

```r
fslmask(
  file,
  mask,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)
```

**Arguments**

- `...`  additional arguments passed to `readnii`
- `outfile` (character) resultant masked image name
- `retimg` (logical) return image of class `nifti`
- `file` (character) image to be masked
- `mask` (character) mask given for image
- `reorient` (logical) If `retimg`, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical) to be passed to `system`
- `opts` (character) additional options to be passed to `fslmask`
- `verbose` (logical) print out command before running
Value

Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

Examples

```r
if (have.fsl()){
  system.time(
    x = array(rnorm(1e5), dim = c(100, 100, 10))
    img = nifti(x, dim= c(100, 100, 10),
    datatype = convert.datatype()$FLOAT32, cal.min = min(x),
    cal.max = max(x), pixdim = rep(1, 4))
    mask = img > .5
    masked = fslmask(img, mask = mask, retimg=TRUE)
  }
}
```

**fsl_maths**

### Description

This function calls fslmaths

### Usage

```r
fsl_maths(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fslmaths(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)
```

#### Arguments

- `...` additional arguments passed to `readnii`
- `outfile` (character) resultant image name (optional)
\textit{fsl_merge} \hfill 103

- \texttt{retimg} (logical) return image of class nifti
- \texttt{file} (character) image to be manipulated
- \texttt{reorient} (logical) If \texttt{retimg}, should file be reoriented when read in? Passed to \texttt{readnii}.
- \texttt{intern} (logical) to be passed to \texttt{system}
- \texttt{opts} (character) operations to be passed to \texttt{fslmaths}
- \texttt{verbose} (logical) print out command before running

\textbf{Value}

If \texttt{retimg} then object of class nifti. Otherwise, Result from system command, depends if \texttt{intern} is \texttt{TRUE} or \texttt{FALSE}.

\textbf{Note}

Functions with underscores have different defaults and will return an output filename, so to be used for piping.

---

\textbf{fsl_merge} \hfill \textit{Merge images using FSL}

\textbf{Description}

This function calls \texttt{fslmerge} to merge files on some dimension and either saves the image or returns an object of class nifti

\textbf{Usage}

\begin{verbatim}
fsl_merge(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslmerge(
  infiles,
  direction = c("x", "y", "z", "t", "a"),
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  verbose = TRUE,
  ...
)
\end{verbatim}

\textbf{Arguments}

- \texttt{...} additional arguments passed to \texttt{readnii}.
- \texttt{outfile} (character) output filename
- \texttt{retimg} (logical) return image of class nifti
fsl_mul

Multiply Images using FSL

Description
This function calls fslmaths -mul. The R functions wraps fslmaths

Usage

fsl_mul(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslmul(
  file,
  file2,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) input image
file2 (character) image to be multiplied
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths
Value

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

**fsl_nan**

*Replace NaNs in Image using FSL*

---

**Description**

This function calls `fslmaths -nan`. The R functions wraps `fslmaths`

**Usage**

```r
fsl_nan(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
def fslnan(
    file,
    outfile = NULL,
    retimg = TRUE,
    reorient = FALSE,
    intern = FALSE,
    opts = "",
    ...
)
```

**Arguments**

- `...`: additional arguments passed to `readnii`
- `outfile`: (character) resultant image name (optional)
- `retimg`: (logical) return image of class nifti
- `file`: (character) input image to replace NaNs (improper numbers) with 0
- `reorient`: (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- `intern`: (logical) to be passed to `system`
- `opts`: (character) operations to be passed to `fslmaths`

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.
Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping.

---

**fsl_nanm**

Mask NaNs in Image using FSL

---

**Description**

This function calls fslmaths -nanm. The R functions wraps fslmaths

**Usage**

```r
fsl_nanm(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
fslnanm(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
```

**Arguments**

- `...` additional arguments passed to `readnii`
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `file` (character) input image to set to 1 for NaN voxels, 0 otherwise
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical) to be passed to `system`
- `opts` (character) operations to be passed to `fslmaths`

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

**Note**

Functions with underscores have different defaults and will return an output filename, so to be used for piping.
Add Random Uniform Noise Image using FSL

Description
This function calls fslmaths -rand. The R functions wraps fslmaths.

Usage
fsl_rand(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslrand(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments

... additional arguments passed to readnii.

outfile (character) resultant image name (optional)

retimg (logical) return image of class nifti

file (character) input image to add random uniform noise to

reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.

intern (logical) to be passed to system

opts (character) operations to be passed to fslmaths

Value
If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note
Functions with underscores have different defaults and will return an output filename, so to be used for piping.
fsl_randn

Add Random Standard Gaussian Noise Image using FSL

Description
This function calls fslmaths -randn. The R functions wraps fslmaths

Usage
fsl_randn(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslrandn(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments
... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) input image to add random standard to Gaussian noise
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths

Value
If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note
Functions with underscores have different defaults and will return an output filename, so to be used for piping
fsl_recip  

Reciprocal Image using FSL

Description

This function calls fslmaths -recip. The R functions wraps fslmaths.

Usage

fsl_recip(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
fslrecip(  
  file,  
  outfile = NULL,  
  retimg = TRUE,  
  reorient = FALSE,  
  intern = FALSE,  
  opts = "",  
  ...
)

Arguments

...  additional arguments passed to readnii.
outfile  (character) resultant image name (optional)
retimg  (logical) return image of class nifti
file  (character) input image to take the reciprocal (1/image)
reorient  (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern  (logical) to be passed to system
opts  (character) operations to be passed to fslmaths

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping.
Modulus Remainder of 2 Images using FSL

Description

This function calls fslmaths -rem. The R functions wraps fslmaths

Usage

fsl_rem(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslrem(
  file,
  file2,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) input image
file2 (character) image to divide the current image by and take remainder
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping
Resample an Image to Specific Voxel Size

Description

Resample an Image to Specific Voxel Size

Usage

fsl_resample(
  file,
  voxel_size,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  opts = NULL,
  verbose = TRUE
)

Arguments

- **file**: Input file to resample
- **voxel_size**: Voxel size (in mm). This should be a scalar number.
- **outfile**: (character) output filename
- **retimg**: (logical) return image of class nifti
- **reorient**: (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- **opts**: options to pass to flirt
- **verbose**: (logical) print out command before running

Value

If retimg then object of class nifti. Otherwise, the output file.

Examples

```r
if (have_fsl()) {
  file = mni_fname(mm = 1, brain = TRUE)
  est2 = fsl_resample(file = file, voxel_size = 1, retimg = FALSE)
  pixdim(est2)
  est = fsl_resample(file = file, voxel_size = 1)
  pixdim(est)
}
```
Description

This function calls fslmaths -s to smooth an image and either saves the image or returns an object of class nifti.

Usage

fsl_smooth(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslsmooth(
  file,
  sigma = 10,
  mask = NULL,
  smooth_mask = TRUE,
  smoothed_mask = NULL,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  verbose = TRUE,
  ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant smoothed image name (optional) if not give, will be the stub of the filename then _sigma
retimg (logical) return image of class nifti
file (character or nifti) image to be smoothed
sigma (numeric) sigma (in mm) of Gaussian kernel for smoothing
mask (character) optional mask given for image
smooth_mask (logical) Smooth mask? If TRUE, the masked image will be divided by the smoothed mask.
smoothed_mask (character or nifti) If specified and smooth_mask = TRUE, then will use this as the smoothed mask for division.
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
verbose (logical) print out command before running
Value

Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping.

Examples

```r
if (have.fsl()){
  system.time({
    dims = c(50, 50, 20)
    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))
    s.img = fslsmooth(img, retimg=TRUE)
  })
}
```

---

**fsl_smoothest**

Smoothness Estimation using smoothest

**Description**

Smoothness Estimation using smoothest

**Usage**

```r
fsl_smoothest(
  file,
  residual_image,
  z_image,
  dof = NULL,
  opts = "",
  verbose = TRUE,
  ...
)
```

**Arguments**

- **file**: filename of input brain mask
- **residual_image**: 4d residual image. If specified, then dof must be specified.
- **z_image**: z-statistic image. Cannot be specified if residual_image is specified
- **dof**: number of degrees of freedom
- **opts**: (character) operations to be passed to smoothest
verbose (logical) print out command before running

... additional arguments to pass to \texttt{fslcmd}

\textbf{Value}

An output of smoothness estimate

\textbf{Examples}

\begin{verbatim}
if (have_fsl()) {
  file = mni_fname(mm = 2, brain = TRUE, mask = TRUE)
  img = mni_img(mm = 2, brain = TRUE, mask = FALSE)
  mask = mni_img(mm = 2, brain = TRUE, mask = TRUE)
  img = zscore_img(img = img, mask = mask)
  est = fsl_smoothest(file = file, z_image = img)
}
\end{verbatim}

\textit{fsl_sqr} \hspace{1cm} \textit{Square Image using FSL}

\textbf{Description}

This function calls \texttt{fslmaths -sqr}. The R functions wraps \texttt{fslmaths}

\textbf{Usage}

\begin{verbatim}
fsl_sqr(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslsqr(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  ...
)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{...} additional arguments passed to \texttt{readnii}.
  \item \texttt{outfile} (character) resultant image name (optional)
  \item \texttt{retimg} (logical) return image of class nifti
  \item \texttt{file} (character) input image to square
  \item \texttt{reorient} (logical) If retimg, should file be reoriented when read in? Passed to \texttt{readnii}.
  \item \texttt{intern} (logical) to be passed to \texttt{system}
  \item \texttt{opts} (character) operations to be passed to \texttt{fslmaths}
\end{itemize}
**fsl_sqrt**

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.

**Note**

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

**fsl_sqrt**

Square Root Image using FSL

**Description**

This function calls `fslmaths -sqrt`. The R functions wraps `fslmaths`

**Usage**

```r
fsl_sqrt(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
def fslsqrt(
    file,
    outfile = NULL,
    retimg = TRUE,
    reorient = FALSE,
    intern = FALSE,
    opts = "",
    ...
)
```

**Arguments**

- `...` additional arguments passed to `readnii`.
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `file` (character) input image to square root
- `reorient` (logical) If `retimg`, should file be reoriented when read in? Passed to `readnii`
- `intern` (logical) to be passed to `system`
- `opts` (character) operations to be passed to `fslmaths`

**Value**

If `retimg` then object of class nifti. Otherwise, Result from system command, depends if `intern` is TRUE or FALSE.
Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping.

---

fsl_std_dir

Get FSL's Standard Data Directory

Description

Finds the FSLDIR from system environment or getOption("fsl.path") and pastes on “data/standard”

Usage

fsl_std_dir()

fsl_std_file(file = NULL)

Arguments

file

A file from the standard data file

Value

Character path

---

fsl_sub

Subtract Images using FSL

Description

This function calls fslmaths -sub. The R functions wraps fslmaths

Usage

fsl_sub(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslsub(
    file,
    file2,
    outfile = NULL,
    retimg = TRUE,
    reorient = FALSE,
    intern = FALSE,
    opts = ",",
    ...
)

}
Arguments

... additional arguments passed to `readnii`.
outfile (character) resultant image name (optional)
reorient (logical) return image of class nifti
file (character) input image
file2 (character) image to be subtracted
reorient (logical) If reorient, should file be reoriented when read in? Passed to `readnii`.
intern (logical) to be passed to `system`
.opts (character) operations to be passed to `fslmaths`

Value

If `reorient` then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

fsl_sub2 Subsample image by factor of 2

Description

This function calls `fslmaths -subsamp2` to subsample an image and either saves the image or returns an object of class nifti

Usage

```r
fsl_sub2(..., outfile = tempfile(fileext = "nii.gz"), reorient = FALSE)
```

```r
fslsub2(
  file,
  outfile = NULL,
  reorient = TRUE,
 Intern = FALSE,
  verbose = TRUE,
  ...)
```

fsl_swapdim

**fsl_swapdim**

**FSL Swap Dimensions**

**Description**

This function calls fslswapdim.

**Usage**

```r
fsl_swapdim(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
```

```r
def fslswapdim(
    file,
    outfile = NULL,
    retimg = TRUE,
```
Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) image to be manipulated
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
a (character) Option for x domain in fslswapdim
b (character) Option for y domain in fslswapdim
c (character) Option for z domain in fslswapdim
verbose (logical) print out command before running

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

fsl_tan

Tangent Transform Image using FSL

Description

This function calls fslmaths -tan. The R functions wraps fslmaths
Usage

fsl_tan(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fsltan(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = ",",
  ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
file (character) input image to tangent transform
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
opts (character) operations to be passed to fslmaths

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

---

fsl_thresh

Threshold an image

Description

This function calls fslmaths -thr -uthr to threshold an image and either saves the image or returns an object of class nifti
Usage

fsl_thresh(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)

fslthresh(
  file,
  outfile = NULL,
  thresh = 0,
  uthresh = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)

Arguments

... additional arguments passed to readnii.
outfile (character) name of resultant thresholded file
retimg (logical) return image of class nifti
file (character) filename of image to be thresholded
thresh (numeric) threshold (anything below set to 0)
uthresh (numeric) upper threshold (anything above set to 0)
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) pass to system
opts (character) additional options to be passed to fslmaths
verbose (logical) print out command before running

Value

character or logical depending on intern

Note

Functions with underscores have different defaults and will return an output filename, so to be used for piping

Examples

if (have.fsl()){
  system.time(
    x = array(rnorm(1e6), dim = c(100, 100, 100))
    img = nifti(x, dim= c(100, 100, 100),
      datatype = convert.datatype()$FLOAT32, cal.min = min(x),
      cal.max = max(x), pixdim = rep(1, 4))
    thresh = fslthresh(img, thresh=0, uthresh = 2, retimg=TRUE)
  )
}
**fsl_tsplot**  
*FSL Timeseries Plot using ‘fsl_tsplot’ (not ‘tsplot’)*

**Description**

FSL Timeseries Plot using ‘fsl_tsplot’ (not ‘tsplot’)

**Usage**

```r
fsl_tsplot(
  infile, 
  outfile = tempfile(fileext = ".png"),
  plot_title = NULL,
  legend = NULL,
  labels = NULL,
  ymin = NULL,
  ymax = NULL,
  xlabel = NULL,
  ylabel = NULL,
  height = NULL,
  width = NULL,
  precision = NULL,
  unit = NULL,
  scientific_notation = FALSE,
  start_position = NULL,
  end_position = NULL,
  ...
)
```

**Arguments**

- **infile**: comma-separated list of input file names (ASCII text matrix, one column per timecourse)
- **outfile**: output filename for the PNG file
- **plot_title**: plot title
- **legend**: file name of ASCII text file, one row per legend entry
- **labels**: comma-separated list of labels
- **ymin**: minimum y-value
- **ymax**: maximum y-value
- **xlabel**: X-axis label
fsl_version

ylabel

height

width

precision

unit

scientific_notation

start_position

end_position

... additional options to pass to fslcmd

Value

Name of PNG file

Description

Finds the FSL version from FSLDIR/etc/fslversion

Usage

fsl_version(full = FALSE)

fslversion()

fsl_version_gt5()

Arguments

full provide the full version, versus the numeric version

Value

If the version file does not exist, it will throw a warning, but it will return an empty string. Otherwise it will be a string of the version.

Note

This will use fsldir() to get the directory
Examples

```r
if (have_fsl()) {
  fslversion()
  fsl_version()
}
```

---

**get.fsl**

*Create command declaring FSLDIR*

---

**Description**

Finds the FSLDIR from system environment or `getOption("fsl.path")` for location of FSL functions.

**Usage**

```r
get.fsl(add_bin = TRUE)
get_fsl(add_bin = TRUE)
```

**Arguments**

- `add_bin`:
  Should bin be added to the fsl path? All executables are assumed to be in `FSLDIR/bin/`. If not, and `add_bin = FALSE`, they will be assumed to be in `FSLDIR/`.

**Value**

NULL if FSL in path, or bash code for setting up FSL DIR

**Note**

This will use `Sys.getenv("FSLDIR")` before `getOption("fsl.path")`. If the directory is not found for FSL in `Sys.getenv("FSLDIR")` and `getOption("fsl.path")`, it will try the default directory `/usr/local/fsl`. 
get.fsloutput  

*Description*

Finds the FSLOUTPUTTYPE from system environment or `getOption("fsl.outputtype")` for output type (nii.gz, nii, ANALYZE, etc).

*Usage*

```r
get.fsloutput()
```

*Value*

FSLOUTPUTTYPE, such as NIFTI_GZ. If none found, uses NIFTI_GZ as default.

---

get.imgext  

*Description*

Runs `get.fsloutput()` to extract FSLOUTPUTTYPE and then gets corresponding extension (such as .nii.gz).

*Usage*

```r
get.imgext()
```

*Value*

Extension for output type
get_forms

Get Q and S Forms of orientation matrix

**Description**

This function obtains the s and q forms of an image transformation matrix.

**Usage**

```r
getForms(file, verbose = FALSE, ...)
```

**Arguments**

- `file` (character) filename of image to pass to header
- `verbose` (logical) passed to `fslhd`
- `...` options passed to `checkimg`

**Value**

list with elements of sform and qform and their respective codes

**Examples**

```r
if (have.fsl()){
  mnifile = mni_fname("2")
  getForms(mnifile)
}
```

---

get_quickshear_mask

Face Removal Mask using "Quickshear Defacing for Neuroimages" (Schimke et al. 2011)

**Description**

Face Removal Mask using "Quickshear Defacing for Neuroimages" (Schimke et al. 2011)

**Usage**

```r
get_quickshear_mask(brain_mask, buffer = 10, verbose = TRUE)
```

```r
quickshear_deface_image(
  file,
  brain_mask = NULL,
  buffer = 10,
  verbose = TRUE,
  ...
)
```
arguments

brain_mask  Brain mask image. If NULL, then fslbet will be run
buffer  buffer to add to intercept for face mask equation
verbose  print diagnostic messages
file  input image - same orientation as brain mask
...  additional arguments passed to fslmask

value

A binary image of the non-face areas

note

adapted from https://github.com/nipy/quickshear/blob/master/quickshear.py

examples

if (have_fsl()) {
  file = "~/Downloads/sample_T1_input.nii.gz"
  if (file.exists(file)) {
    res = quickshear_deface_image(file)
    brain_mask = fslbet(file) > 0
    mask = get_quickshear_mask(brain_mask)
    image = fslmask(file, mask)
  }
}

have.fsl  Logical check if FSL is accessible

description

Uses get.fsl to check if FSLDIR is accessible or the option fsl.path is set and returns logical

usage

have.fsl(...)  

have_fsl(...)  

arguments

...  options to pass to get.fsl
Value
Logical TRUE is FSL is accessible, FALSE if not

Examples
have.fsl()

intent_code-methods  
Extract Image intent_code attribute

Description
intent_code method for character types

Usage
## S4 method for signature 'character'
intent_code(object)

Arguments
object is a filename to pass to fslval

intent_name-methods  
Extract Image intent_name attribute

Description
intent_name method for character types

Usage
## S4 method for signature 'character'
intent_name(object)

Arguments
object is a filename to pass to fslval
### Description

intent_p1 method for character types

### Usage

```r
## S4 method for signature 'character'
intent_p1(object)
```

### Arguments

- `object` is a filename to pass to `fslval`

### Description

intent_p2 method for character types

### Usage

```r
## S4 method for signature 'character'
intent_p2(object)
```

### Arguments

- `object` is a filename to pass to `fslval`

### Description

intent_p3 method for character types

### Usage

```r
## S4 method for signature 'character'
intent_p3(object)
```

### Arguments

- `object` is a filename to pass to `fslval`
invert_xfm

Convert a Transformation

Description

Convert a Transformation

Usage

invert_xfm(inmat, omat = tempfile(fileext = ".mat"), verbose = TRUE)

concat_xfm(inmat, inmat2, omat = tempfile(fileext = ".mat"), verbose = TRUE)

fixscaleskew_xfm(
    inmat,
    inmat2,
    omat = tempfile(fileext = ".mat"),
    verbose = TRUE
)

Arguments

inmat  input matrix transformation
omat   output matrix transformation
verbose print diagnostic messages
inmat2  second matrix filename to be concatenated or fixscaleskew to first

Value

A filename of the output matrix file

Examples

if (have_fsl()) {
    img = mni_fname()
    mat = fslreorient2std_mat(img)
    inverted = invert_xfm(mat)
    readLines(inverted)
    catted = concat_xfm(mat, mat)
    readLines(catted)
    fixed = fixscaleskew_xfm(mat, mat)
    readLines(fixed)
}
**magic-methods**

Extract Image magic attribute

**Description**

magic method for character types

**Usage**

```r
## S4 method for signature 'character'
magic(object)
```

**Arguments**

- **object** is a filename to pass to `fslval`

**mcflirt**

*FSL Motion Correction*

**Description**

This function calls `mcflirt`

**Usage**

```r
mcflirt(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)
```

**Arguments**

- **file** (character) image to be manipulated
- **outfile** (character) resultant image name (optional)
- **retimg** (logical) return image of class nifti
- **reorient** (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
- **intern** (logical) to be passed to `system`
- **opts** (character) operations to be passed to `mcflirt`. Cannot use `-o` or `-verbose`, as output file should be specified in outfile.
- **verbose** (logical) print out command before running
- **...** additional arguments passed to `readnii`
melodic

Value

If `retimg` then object of class `nifti`. Otherwise, it will have additional attributes in the `additional_files` field.

---

mcflirt.help

`MCFLIRT help`

---

Description

This function calls `mcflirt`'s help

Usage

`mcflirt.help()`

Value

Prints help output and returns output as character vector

Examples

```r
library(fslr)
if (have.fsl()){ mcflirt.help() }
```

---

melodic

`Run MELODIC ICA`

---

Description

This function calls `melodic`

Usage

```r
melodic(
  file,
  outdir = dirname(file),
  intern = FALSE,
  opts = "",
  verbose = TRUE,
  ...
)
```
Arguments

file (character) image to be run
outdir (character) output directory. (Default dirname(file))
intern (logical) pass to system
opts (character) options for melodic
verbose (logical) print out command before running
... arguments passed to checkimg

Value

character or logical depending on intern

Description

This function calls melodic’s help

Usage

melodic.help()

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  melodic.help()
}
mid_sagittal_align  

**Mid-Sagittal Plane Alignment**

**Description**

This function takes in an image, flips the image over the left/right plane, registers that flipped image to the original image, then applies the half transformation.

**Usage**

```r
mid_sagittal_align(
  file,
  outfile = NULL,
  retimg = TRUE,
  opts = "",
  translation = TRUE,
  force_rpi = TRUE,
  verbose = TRUE
)
```

**Arguments**

- `file` (character): input filename or class nifti
- `outfile` (character): output filename
- `retimg` (logical): return image of class nifti
- `opts` (character): options passed to `flirt`
- `translation` (logical): should the translation parameters be preserved (TRUE) or set to zero (FALSE)
- `force_rpi` (logical): Should `rpi_orient_file` be run?
- `verbose` (logical): print diagnostic messages

**Value**

Filename of output or nifti depending on `retimg`

---

mni_fname  

**Construct MNI Filename**

**Description**

Finds the standard data directory for FSL and pastes together the string for an MNI template image.
Usage

\texttt{mni\_fname(mm = c("1", "0.5", "2"), brain = FALSE, linear = FALSE, mask = FALSE)}

\texttt{mni\_face\_fname(mm = c("1", "0.5", "2"))}

Arguments

\begin{itemize}
  \item \texttt{mm} \hspace{1cm} Resolution (in mm) of the brain image (isotropic)
  \item \texttt{brain} \hspace{1cm} Should the brain be returned (default) or the T1 with the skull
  \item \texttt{linear} \hspace{1cm} Should the linearized MNI template be used
  \item \texttt{mask} \hspace{1cm} should the mask be given? Generally, only MNI152\_T1\_1mm\_brain\_mask exists.
\end{itemize}

Value

Character path of filename, warning if that file does not exist

\textbf{Description}

Simple wrapper for reading in the MNI image constructed from \texttt{mni\_fname}

Usage

\texttt{mni\_img(...)}

Arguments

\begin{itemize}
  \item \texttt{...} \hspace{1cm} Arguments passed to \texttt{mni\_fname}
\end{itemize}

Value

Object of class \texttt{nifti}
**Description**

MRI Defacer

**Usage**

```r
mridefacer(file, ..., verbose = TRUE)

get_mridefacer_mask(
  file,
  brain_mask = NULL,
  bet_opts = "-f 0.5",
  search_radius = 90,
  opts = NULL,
  template_brain = NULL,
  template_brain_weight = NULL,
  template_biometric_mask = NULL,
  verbose = TRUE
)
```

**Arguments**

- `file`: input file image to remove face/ears
- `...`: not used
- `verbose`: print diagnostic messages. If > 1, more verbose
- `brain_mask`: brain mask of file. If NULL, `fslbet` will be applied
- `bet_opts`: options to pass to `fslbet` if applied
- `search_radius`: search radius option to pass to `flirt`
- `opts`: additional options to pass to `flirt`
- `template_brain`: template brain image, may be NULL
- `template_brain_weight`: template brain weight image, used for registration may be NULL
- `template_biometric_mask`: template biometric mask. Everything that is wanted should be 1, may be NULL

**Value**

A character filename of the output image

**Note**

Adapted from https://github.com/mih/mridefacer
Examples
if (have_fsl()) {
  file = "~/Downloads/sample_T1_input.nii.gz"
  if (file.exists(file)) {
    res = mridefacer(file)
  }
}

parse_avscale
parse_avscale      Parse output from avscale

Description
This function parses the output from \texttt{fsl_avscale} into something more manageable

Usage
parse_avscale(av_out)

Arguments
  av_out          output from \texttt{fsl_avscale}, character vector

Value
List of output values

pixdim-methods
pixdim-methods      Extract Image pixdim attribute

Description
Gets pixdim from a character

Usage
## S4 method for signature 'character'
pixdim(object)

Arguments
  object       is a filename to pass to \texttt{fslval}
probtrackx  

Probabilistic diffusion tractography with multiple fibre orientations

Description

This function wraps probtrackx from FSL

Usage

probtrackx(
  samples = "merged",
  mask,
  seed,
  outdir = "fdt_paths",
  verbose = TRUE,
  mode = NULL,
  targetmasks = NULL,
  mask2 = NULL,
  waypoints = NULL,
  network = FALSE,
  mesh = NULL,
  seedref = NULL,
  dir = FALSE,
  forcedir = FALSE,
  opd = FALSE,
  pd = FALSE,
  os2t = FALSE,
  avoid = NULL,
  stop = NULL,
  xfm = NULL,
  invxfm = NULL,
  nsamples = 5000,
  nsteps = 2000,
  distthresh = 0,
  cthr = 0.2,
  fibthresh = 0.01,
  sampvox = FALSE,
  steplength = 0.5,
  loopcheck = FALSE,
  usef = FALSE,
  randfib = c(0, 1, 2, 3),
  fibst = 1,
  modeuler = FALSE,
  rseed = NULL,
  s2tastext = FALSE,
  opts = ""
)
### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>samples</td>
<td>Basename for samples files</td>
</tr>
<tr>
<td>mask</td>
<td>Bet binary mask file in diffusion space</td>
</tr>
<tr>
<td>seed</td>
<td>Seed volume, or voxel, or ascii file with multiple volumes, or freesurfer label file</td>
</tr>
<tr>
<td>outdir</td>
<td>Output file (default='fdt_paths')</td>
</tr>
<tr>
<td>verbose</td>
<td>Verbose level, [0-2]</td>
</tr>
<tr>
<td>mode</td>
<td>Use –mode=simple for single seed voxel</td>
</tr>
<tr>
<td>targetmasks</td>
<td>File containing a list of target masks - required for seeds_to_targets classification</td>
</tr>
<tr>
<td>mask2</td>
<td>Second mask in twomask_symm mode.</td>
</tr>
<tr>
<td>waypoints</td>
<td>Waypoint mask or ascii list of waypoint masks - only keep paths going through ALL the masks</td>
</tr>
<tr>
<td>network</td>
<td>Activate network mode - only keep paths going through at least one seed mask (required if multiple seed masks)</td>
</tr>
<tr>
<td>mesh</td>
<td>Freesurfer-type surface descriptor (in ascii format)</td>
</tr>
<tr>
<td>seedref</td>
<td>Reference vol to define seed space in simple mode - diffusion space assumed if absent</td>
</tr>
<tr>
<td>dir</td>
<td>Directory to put the final volumes in - code makes this directory - default='logdir'</td>
</tr>
<tr>
<td>forcedir</td>
<td>Use the actual directory name given - i.e. don’t add + to make a new directory</td>
</tr>
<tr>
<td>opd</td>
<td>Output path distribution</td>
</tr>
<tr>
<td>pd</td>
<td>Correct path distribution for the length of the pathways</td>
</tr>
<tr>
<td>os2t</td>
<td>Output seeds to targets</td>
</tr>
<tr>
<td>avoid</td>
<td>Reject pathways passing through locations given by this mask</td>
</tr>
<tr>
<td>stop</td>
<td>Stop tracking at locations given by this mask file</td>
</tr>
<tr>
<td>xfm</td>
<td>Transform taking seed space to DTI space (either FLIRT matrix or FNIRT warpfield) - default is identity</td>
</tr>
<tr>
<td>invxfm</td>
<td>Transform taking DTI space to seed space (compulsory when using a warpfield for seeds_to_dti)</td>
</tr>
<tr>
<td>nsamples</td>
<td>Number of samples - default=5000</td>
</tr>
<tr>
<td>nsteps</td>
<td>Number of steps per sample - default=2000</td>
</tr>
<tr>
<td>distthresh</td>
<td>Discards samples shorter than this threshold (in mm - default=0)</td>
</tr>
<tr>
<td>cthr</td>
<td>Curvature threshold - default=0.2</td>
</tr>
<tr>
<td>fibthresh</td>
<td>Volume fraction before subsidiary fibre orientations are considered - default=0.01</td>
</tr>
<tr>
<td>sampvox</td>
<td>Sample random points within seed voxels</td>
</tr>
<tr>
<td>steplength</td>
<td>Steplength in mm - default=0.5</td>
</tr>
</tbody>
</table>
loopcheck (logical) Perform loopchecks on paths - slower, but allows lower curvature threshold

usef (logical) Use anisotropy to constrain tracking

randfib (numeric) Default 0. Set to 1 to randomly sample initial fibres (with f > fibthresh). Set to 2 to sample in proportion fibres (with f>fibthresh) to f. Set to 3 to sample ALL populations at random (even if f<fibthresh)

fibst (numeric) Force a starting fibre for tracking - default=1, i.e. first fibre orientation. Only works if randfib==0

modeuler (logical) Use modified euler streamlining

rseed (numeric) Random seed

s2tastext (logical) Output seed-to-target counts as a text file (useful when seeding from a mesh)

opts Additional options or way to specify things instead of command line arguments

Value
A filename of the output file

qform,character-method

Extract NIfTI 3D Image Orientation

Description
Gets q/s-forms from a character

Usage

## S4 method for signature 'character'
qform(object)

## S4 method for signature 'character'
sform(object)

Arguments

object is a nifti object
qform_code-methods

### Description

qform_code method for character types

### Usage

```r
## S4 method for signature 'character'
qform_code(object)
```

### Arguments

- `object` is a filename to pass to `fslval`

---

### readrpi

**Read NiFTI file reoriented to RPI**

### Description

This function calls the `readnii` function after calling `rpi_orient_file` to force RPI orientation.

### Usage

```r
readrpi(file, ..., verbose = TRUE)
```

### Arguments

- `file` file name of the NiFTI file.
- `...` Arguments to pass to `readnii`
- `verbose` print diagnostics, passed to `rpi_orient_file`

### Examples

```r
if (have.fsl()){
  print(fsl_version())
  in_ci <- function() {
    nzchar(Sys.getenv("CI"))
  }
  if (in_ci()) {
    destfile = tempfile(fileext = ".nii.gz")
    url = paste0("https://ndownloader.figshare.com/",
                 "files/18068546")
    old_url = paste0("https://github.com/muschellij2/",
                     "Neurohacking/files/3454385/113-01-MPRAGE2.nii.gz")
```
dl = tryCatch(download.file(url, 
destfile = destfile))
if (inherits(dl, "try-error") || dl != 0) {
  dl = download.file(old_url, destfile = destfile)
}
res = readrpi(destfile)
)
)

read_xfm  Read FSL Transformation

Description
Read FSL Transformation

Usage
read_xfm(file)

Arguments
file  transformation file from flirt, usually ending in `.mat`

Value
A numeric matrix of numeric class

reverse_rpi_orient  Reverse Reorientation an Image to RPI orientation

Description
This function uses fslswapdim to reorient an image

Usage
reverse_rpi_orient( 
  file, 
  convention = c("NEUROLOGICAL", "RADIOLOGICAL"), 
  orientation, 
  verbose = TRUE 
)

reverse_rpi_orient_file( 
  file,
rpi_orient

```
convention = c("NEUROLOGICAL", "RADIOLOGICAL"),
orientation,
verbose = TRUE
)
```

Arguments

- **file**: Object of class nifti or character path
- **convention**: Convention of original image (usually from `rpi_orient`)
- **orientation**: Vector of length 3 from original image (usually from `rpi_orient`)
- **verbose**: print diagnostic messages

Value

Object of class nifti

---

**rpi_orient**  
*Reorient an Image to RPI orientation*

Description

This function uses `fslswapdim` to reorient an image

Usage

- `rpi_orient(file, verbose = TRUE)`
- `rpi_orient_file(file, verbose = TRUE)`
- `is_rpi(file, verbose = FALSE)`
- `is.rpi(file, verbose = FALSE)`

Arguments

- **file**: Object of class nifti or character path
- **verbose**: print diagnostic messages

Value

List of 3 elements

- **img**: Reoriented image of class nifti
- **convention**: Convention (Neurological/Radiological) of original image
- **orientation**: Original image orientations
Note

`orient_rpi` and `orient_rpi_file` use `RNifti` to ensure the reading orientation.

Examples

```r
lr_fname = system.file( "nifti", "mniLR.nii.gz", package = "oro.nifti")
img = readnii(lr_fname)

rl_fname = system.file( "nifti", "mniRL.nii.gz", package = "oro.nifti")
rl_img = readnii(rl_fname)
stopifnot(all(rl_img[nrow(rl_img):1,,] == img))

## Not run:
if (have_fsl()) {

reor = rpi_orient(rl fname)
rev = reverse_rpi_orient(reor$img, convention = reor$convention,
orientation = reor$orientation)
stopifnot(all(rev == rl_img))
}

## End(Not run)

reor = orient_rpi(rl_fname)
stopifnot(all(img == reor$img))

rev = reverse_orient_rpi(reor$img, convention = reor$convention,
orientation = reor$orientation)
stopifnot(all(rev == rl_img))
```

---

**run_first_all**  
**Run FIRST All**

Description

Wrapper for `run_first_all` from FSL for FIRST analysis segmentation of subcortical structures

Usage

```r
run_first_all(
  img,
  oprefix = tempfile(),
  brain_extracted = FALSE,
  structures = NULL,
  affine = NULL,
  opts = "",
  verbose = TRUE
)
```
run_first_all.help

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>img</td>
<td>specifies the input image (T1-weighted)</td>
</tr>
<tr>
<td>oprefix</td>
<td>specifies the output image basename (extensions will be added to this)</td>
</tr>
<tr>
<td>brain_extracted</td>
<td>specifies that the input image has been brain extracted</td>
</tr>
<tr>
<td>structures</td>
<td>a restricted set of structures to be segmented</td>
</tr>
<tr>
<td>affine</td>
<td>specifies the affine registration matrix to standard space (optional)</td>
</tr>
<tr>
<td>opts</td>
<td>(character) operations to be passed to run_first_all</td>
</tr>
<tr>
<td>verbose</td>
<td>(logical) print out command before running</td>
</tr>
</tbody>
</table>

Value

List of results, including result of `system` and some output files

---

**Description**

This function calls `run_first_all`'s help

**Usage**

`run_first_all.help()`

**Value**

Prints help output and returns output as character vector

**Examples**

```r
code
library(fslr)
if (have.fsl()){
  run_first_all.help()
}
```
scl_inter-methods

Description

scl_inter method for character types

Usage

## S4 method for signature 'character'
scl_inter(object)

Arguments

object is a filename to pass to fslval

scl_slope-methods

Description

scl_slope method for character types

Usage

## S4 method for signature 'character'
scl_slope(object)

Arguments

object is a filename to pass to fslval

sform_code-methods

Description

sform_code method for character types

Usage

## S4 method for signature 'character'
sform_code(object)

Arguments

object is a filename to pass to fslval
sizeof_hdr-methods

**Description**

'sizeof_hdr' method for character types

**Usage**

```r
## S4 method for signature 'character'
sizeof_hdr(object)
```

**Arguments**

- `object` is a filename to pass to `fslval`

slice_code-methods

**Description**

slice_code method for character types

**Usage**

```r
## S4 method for signature 'character'
slice_code(object)
```

**Arguments**

- `object` is a filename to pass to `fslval`
slice_duration-methods

Extract Image slice_duration attribute

Description

slice_duration method for character types

Usage

## S4 method for signature 'character'
slice_duration(object)

Arguments

object is a filename to pass to fslval

slice_end-methods

Extract Image slice_end attribute

Description

slice_end method for character types

Usage

## S4 method for signature 'character'
slice_end(object)

Arguments

object is a filename to pass to fslval
Description

slice_start method for character types

Usage

## S4 method for signature 'character'
slice_start(object)

Arguments

object is a filename to pass to fslval

susan  

FSL SUSAN noise reduction

Description

Implements Smallest Univalue Segment Assimilating Nucleus (SUSAN) noise reduction technique from FSL

Usage

susan(
  file,
  outfile = NULL,
  retimg = TRUE,
  reorient = FALSE,
  intern = FALSE,
  bthresh = 0.1,
  sigma = 3,
  dimg = c(3, 2),
  use_median = FALSE,
  n_usans = c(0, 1, 2),
  extra.scans = list(),
  opts = "",
  verbose = TRUE,
  ...)

)
Arguments

file (character) image to be manipulated
outfile (character) resultant image name (optional)
retimg (logical) return image of class nifti
reorient (logical) If retimg, should file be reoriented when read in? Passed to readnii.
intern (logical) to be passed to system
bthresh brightness threshold and should be greater than noise level and less than contrast of edges to be preserved.
sigma spatial size (sigma i.e. half-width) of smoothing in mm.
dimg dimensionality (2 or 3) depending on whether smoothing is to be within-plane (2) or fully 3D (3).
use_median determines whether to use a local median filter in the cases where single-point noise is detected (0 or 1).
n_usans determines whether the smoothing area (USAN) is to be found from secondary images (0 1 or 2).
extra.scans List of extra scans for USAN. List of n_usans elements, where each element has 2 named objects bthresh and filename
opts (character) operations to be passed to susan, not currently used.
verbose (logical) print out command before running
... additional arguments passed to fslcmd.

Value

If retimg then object of class nifti. Otherwise, Result from system command, depends if intern is TRUE or FALSE.

References


Description

This function calls susan’s help

Usage

susan.help()
toffset-methods

Value

Prints help output and returns output as character vector

Examples

```r
library(fslr)
if (have.fsl()){
  susan.help()
}
```

tooffset-methods  Extract Image toffset attribute

Description

Gets toffset from a character

Usage

```r
## S4 method for signature 'character'
tooffset(object)
```

Arguments

- `object` is a filename to pass to `fslval`

Description

topup - calling FSL topup

Usage

```r
topup(
  infile,
  datain,
  out = NULL,
  fout = NULL,
  iout = NULL,
  logout = NULL,
  warpres = 10,
  subsamp = 1,
  fwhm = 8,
  config = NULL,
)```
miter = 5,
lambda = NULL,
ssqlambda = 1,
regmod = c("bending_energy", "membrane_energy"),
estmov = 1,
minmet = c(0, 1),
splineorder = c(3, 2),
numprec = c("double", "float"),
interp = c("spline", "linear"),
scale = c(0, 1),
regrid = c(0, 1),
verbose = TRUE
)

fsl_topup(...)

Arguments

infile  name of 4D file with images
datain   name of text file with PE directions/times
out      base-name of output files (spline coefficients (Hz) and movement parameters)
fout     name of image file with field (Hz)
iout     name of 4D image file with unwarped images
logout   Name of log-file
warpres  (approximate) resolution (in mm) of warp basis for the different sub-sampling levels, default 10
subsamp  sub-sampling scheme, default 1
fwhm     FWHM (in mm) of gaussian smoothing kernel, default 8
config   Name of config file specifying command line arguments
miter    Max # of non-linear iterations, default 5
lambda   Weight of regularisation, default depending on ssqlambda and regmod switches. See user documentation.
ssqlambda If set (=1), lambda is weighted by current ssq, default 1
regmod   Model for regularisation of warp-field [membrane_energy bending_energy], default bending_energy
estmov   Estimate movements if set, default 1 (true)
minmet   Minimisation method 0=Levenberg-Marquardt, 1=Scaled Conjugate Gradient, default 0 (LM)
splineorder Order of spline, 2->Quadratic spline, 3->Cubic spline. Default=3
numprec  Precision for representing Hessian, double or float. Default double
interp   Image interpolation model, linear or spline. Default spline
scale    If set (=1), the images are individually scaled to a common mean, default 0 (false)
### Description

vox_offset method for character types

### Usage

```r
## S4 method for signature 'character'
vox_offset(object)
```

### Arguments

- `object` is a filename to pass to `fslval`

### xfibres

*Bayesian Estimation of Diffusion Parameters Obtained using Sampling Techniques with Crossing Fibers*

### Description

Calls `xfibres` from FSL to fit, also known as `bedpostx`

### Usage

```r
xfibres(
  infile,
  bvecs,
  bvals,
  mask = NULL,
  nfibres = 1,
  bet.opts = "",
  verbose = TRUE,
  njumps = NULL,
  burnin = NULL,
  burnin_noard = NULL,
  sampleevery = NULL,
  updateproposalevery = NULL,
  seed = NULL,
  noard = FALSE,
```
allard = FALSE,
nospat = FALSE,
nonlinear = FALSE,
cnonlinear = FALSE,
rician = FALSE,
f0 = FALSE,
ardf0 = FALSE,
opts = ""
)

Arguments

infile  Input filename
bvecs   b-vectors: matrix of 3 columns or filename of ASCII text file
bvals   b-values: vector of same length as number of rows of b-vectors or filename of
         ASCII text file
mask    Mask filename
nfibres Maximum number of fibres to fit in each voxel (default 1)
bet.opts Options for fslbet if mask is not supplied
verbose print diagnostic messages
njumps  num of jumps to be made by MCMC (default is 5000)
burnin  Total num of jumps at start of MCMC to be discarded (default is 0)
burnin_noard num of burnin jumps before the ard is imposed (default is 0)
sampleevery num of jumps for each sample (MCMC) (default is 1)
updateproposalevery num of jumps for each update to the proposal density std (MCMC) (default is 40)
seed    for pseudo random number generator
noard   Turn ARD off on all fibres
allard  Turn ARD on on all fibres
nospat  Initialise with tensor, not spatially
nonlinear Initialise with nonlinear fitting
cnonlinear Initialise with constrained nonlinear fitting
rician  Use Rician noise modelling
f0      Add to the model an unattenuated signal compartment
ardf0   Use ard on f0
opts    Additional options for xfibres. There should not be any left out in the current
         arguments, but opts may be a way some prefer to input options.

Value

Output from system
Index

apply_topup (applytopup), 6
applytopup, 6
aux.file, character-method
  (aux.file-methods), 7
aux.file-methods, 7
avscale (fsl_avscale), 82
bedpostx (xfibres), 153
bitpix, character-method
  (bitpix-methods), 7
bitpix-methods, 7
cal.max, character-method
  (cal.max-methods), 7
cal.max-methods, 7
cal.min, character-method
  (cal.min-methods), 8
cal.min-methods, 8
cHECK_FILE, 9
cHECKING, 9, 41, 53, 65, 70, 71, 78, 126, 133
cHECKOUT, 8, 9
cONCAT_XFM (inve XFM), 130
data_type, character-method
  (data_type-methods), 10
data_type-methods, 10
datatype, character-method
  (datatype-methods), 9
datatype-methods, 9
deface_image (face_removal_mask), 15
describe, character-method
  (describe-methods), 10
describe-methods, 10
dim, character-method (dim-methods), 10
dim-methods, 10
download_file, 11
download_fsl, 11
dtifit, 11
eddy, 12, 13
eddy_correct, 14
enforce_form, 14
epi_reg (fslepi_reg), 36
face_removal_mask, 15
fast, 16
fast.help, 17
fast_all (fast), 16
fast_nobias (fast), 16
fast_nobias_all (fast), 16
fixscaleskew_xfm (inver XFM), 130
flirt, 18, 19, 134, 136, 142
flirt.help, 19
flirt_apply, 19
flirt_apply.help (flirt.help), 19
fnirt, 20, 23, 79
fnirt.help, 21
fnirt_with_affine, 21
fnirt_with_affine_apply, 22
fsl_abs, 74
fsl_acos, 75
fsl_add, 76
fsl_anat, 77
fsl_anat.help, 78
fsl_and (fsland), 25
fsl_applytopup (applytopup), 6
fsl_applywarp, 79
fsl_applywarp.help, 80
fsl_asin, 80
fsl_atan, 81
fsl_atlas_dir, 82
fsl_avscale, 82, 137
fsl_bet, 83
fsl_biascorrect, 84
fsl_bin, 85
fsl_bin_tab, 87, 92
fsl_binv, 86
fsl_cluster, 87
fsl_cos, 89
fsl_data_dir, 90
fsl_deface, 91
fsl_dice, 92
fsl_dilate, 92
fsl_dir (fsldir), 34
fsl_div, 94
fsl_edge, 95
fsl_epi_reg (fslepi_reg), 36
fsl_erode, 96
fsl_exp, 97
fsl_fast (fast), 16
fsl_fast_nobias (fast), 16
fsl_fill, 98
fsl_index, 99
fsl_log, 100
fsl_mask, 101
fsl_maths, 102
fsl_merge, 103
fsl_mul, 104
fsl_nan, 105
fsl纳米, 106
fsl_or (fslor), 49
fsl_rand, 107
fsl_randn, 108
fsl_recip, 109
fsl_rem, 110
fsl_resample, 111
fsl_robustfov (fslrobustfov), 56
fsl_roi (fslroi), 57
fsl_slicetimer (fslslicetimer), 60
fsl_smooth, 112
fsl_smooth_in_mask (fslsmooth_in_mask), 62
fsl_smoothest, 113
fsl_split (fsslsplit), 63
fsl_sqr, 114
fsl_sqrt, 115
fsl_std_dir, 116
fsl_std_file (fsl_std_dir), 116
fsl_sub, 116
fsl_sub2, 117
fsl_swapdim, 118
fsl_tan, 119
fsl_thresh, 73, 120
fsl_topup (topup), 151
fsl_tpsplot, 122
fsl_version, 123
fsl_version_gt5 (fsl_version), 123
fsl_xor (fslxor), 73

fslabs (fsl_abs), 74
fslabs.help, 24
fslacos (fsl_acos), 75
fslacos.help, 24
fsladd (fsl_add), 76
fsladd.help, 25
fsland, 25
fslasin (fsl_asin), 80
fslasin.help, 26
fslatan (fsl_atan), 81
fslatan.help, 26
fslbet, 12, 127, 136, 154
fslbet (fsl_bet), 83
fslbet.help, 27
fsland, 25, 49, 73
fslbin (fsl_bin), 85
fslbin.help, 28
fslbinv (fsl_binv), 86
fslbinv.help, 28
fslchfiletype, 29
fslchfiletype.help, 30
fslcluster (fsl_cluster), 87
fslcmd, 14, 30, 89, 91, 114, 123, 150
fslcog, 31
fslcos (fsl_cos), 89
fslcos.help, 32
fslcpgeom, 33
fslcpgeom.help, 34
fslcombine (fsl_dilate), 92
fslcombine, 34
fsldivide (fsl_div), 94
fsldivide.help, 35
fsldivide (fsl_edge), 95
fsldivide.help, 35
fslepi_reg, 36
fslepi_reg.help (fslstats.help), 66
fsland, 25
fsland (fsl_or), 49
fsland (fslor), 49
get_mridefacer_mask (mridefacer), 136
get_quickshear_mask, 126
getForms, 8, 14, 126
have.fsl, 127
have_fsl (have.fsl), 127
intent_code, character-method
(intent_code-methods), 128
intent_code-methods, 128
intent_name, character-method
(intent_name-methods), 128
intent_name-methods, 128
intent_p1, character-method
(intent_p1-methods), 129
intent_p1-methods, 129
intent_p2, character-method
(intent_p2-methods), 129
intent_p2-methods, 129
intent_p3, character-method
(intent_p3-methods), 129
intent_p3-methods, 129
invert_xfm, 130
is.rpi (rpi_orient), 143
is_rpi (rpi_orient), 143
magic, character-method (magic-methods), 131
magic-methods, 131
mcflirt, 131
mcflirt.help, 132
melodic, 132
melodic.help, 133
mid_sagittal_align, 134
mni_face_fname, 15
mni_face_fname (mni fname), 134
mni_fname, 15, 134, 135
mni_img, 135
mridefacer, 136
nifti, 135
parse_avscale, 82, 137
pixdim, character-method
(pixdim-methods), 137
pixdim-methods, 137
probtrackx, 138
qform, character
(qform, character-method), 140
scl_inter, character-method
(scl_inter-methods), 146
scl_inter-methods, 146
scl_slope, character-method
(scl_slope-methods), 146
scl_slope-methods, 146
sform, character
(qform, character-method), 140
sform, character-method
(qform, character-method), 140
sform_code, character-method
(sform_code-methods), 146
sform_code-methods, 146
sizeof_hdr, character-method
(sizeof_hdr-methods), 147
sizeof_hdr-methods, 147
slice_code, character-method
(slice_code-methods), 147
slice_code-methods, 147
slice_duration, character-method
(slice_duration-methods), 148
slice_duration-methods, 148
slice_end, character-method
(slice_end-methods), 148
slice_end-methods, 148
read_cluster_table (fsl_cluster), 87
read_xfm, 142
readnii, 17, 18, 20–23, 29, 31, 33, 37, 40, 49,
50, 55, 56, 58, 59, 61, 63, 74–77,
79–81, 83–86, 88, 90, 91, 93–112,
114, 115, 117–121, 131, 141, 150
readdrpi, 141
reverse_rpi_orient, 142
reverse_rpi_orient_file
(reverse_rpi_orient), 142
rpi_orient, 143, 143
rpi_orient_file, 134, 141
rpi_orient_file (rpi_orient), 143
run_first_all, 144
run_first_all.help, 145
scl_inter, character-method
(scl_inter-methods), 146
scl_inter-methods, 146
scl_slope, character-method
(scl_slope-methods), 146
scl_slope-methods, 146
sform, character
(qform, character-method), 140
sform, character-method
(qform, character-method), 140
sform_code, character-method
(sform_code-methods), 146
sform_code-methods, 146
sizeof_hdr, character-method
(sizeof_hdr-methods), 147
sizeof_hdr-methods, 147
slice_code, character-method
(slice_code-methods), 147
slice_code-methods, 147
slice_duration, character-method
(slice_duration-methods), 148
slice_duration-methods, 148
slice_end, character-method
(slice_end-methods), 148
slice_end-methods, 148
get_mridefacer_mask (mridefacer), 136
get_quickshear_mask, 126
getForms, 8, 14, 126
have.fsl, 127
have_fsl (have.fsl), 127
intent_code, character-method
(intent_code-methods), 128
intent_code-methods, 128
intent_name, character-method
(intent_name-methods), 128
intent_name-methods, 128
intent_p1, character-method
(intent_p1-methods), 129
intent_p1-methods, 129
intent_p2, character-method
(intent_p2-methods), 129
intent_p2-methods, 129
intent_p3, character-method
(intent_p3-methods), 129
intent_p3-methods, 129
invert_xfm, 130
is.rpi (rpi_orient), 143
is_rpi (rpi_orient), 143
magic, character-method (magic-methods), 131
magic-methods, 131
mcflirt, 131
mcflirt.help, 132
melodic, 132
melodic.help, 133
mid_sagittal_align, 134
mni_face_fname, 15
mni_face_fname (mni fname), 134
mni_fname, 15, 134, 135
mni_img, 135
mridefacer, 136
nifti, 135
parse_avscale, 82, 137
pixdim, character-method
(pixdim-methods), 137
pixdim-methods, 137
probtrackx, 138
qform, character
(qform, character-method), 140
qform, character-method, 140
qform_code, character-method
(qform_code-methods), 141
qform_code-methods, 141
quickshear_deface_image
(get_quickshear_mask), 126
read_cluster_table (fsl_cluster), 87
read_xfm, 142
readnii, 17, 18, 20–23, 29, 31, 33, 37, 40, 49,
50, 55, 56, 58, 59, 61, 63, 74–77,
79–81, 83–86, 88, 90, 91, 93–112,
114, 115, 117–121, 131, 141, 150
readdrpi, 141
reverse_rpi_orient, 142
reverse_rpi_orient_file
(reverse_rpi_orient), 142
rpi_orient, 143, 143
rpi_orient_file, 134, 141
rpi_orient_file (rpi_orient), 143
run_first_all, 144
run_first_all.help, 145
scl_inter, character-method
(scl_inter-methods), 146
scl_inter-methods, 146
scl_slope, character-method
(scl_slope-methods), 146
scl_slope-methods, 146
sform, character
(qform, character-method), 140
sform, character-method
(qform, character-method), 140
sform_code, character-method
(sform_code-methods), 146
sform_code-methods, 146
sizeof_hdr, character-method
(sizeof_hdr-methods), 147
sizeof_hdr-methods, 147
slice_code, character-method
(slice_code-methods), 147
slice_code-methods, 147
slice_duration, character-method
(slice_duration-methods), 148
slice_duration-methods, 148
slice_end, character-method
(slice_end-methods), 148
slice_end-methods, 148
<table>
<thead>
<tr>
<th>Method</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>slice_start</td>
<td>149</td>
</tr>
<tr>
<td>slice_start-methods</td>
<td>149</td>
</tr>
<tr>
<td>slice_start,character</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>topup</td>
<td>151</td>
</tr>
<tr>
<td>toffset</td>
<td>151</td>
</tr>
<tr>
<td>toffset-methods</td>
<td>151</td>
</tr>
<tr>
<td>toffset,character-method</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>vox_offset</td>
<td>153</td>
</tr>
<tr>
<td>vox_offset-methods</td>
<td>153</td>
</tr>
<tr>
<td>vox_offset,character</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>xfibres</td>
<td>153</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>