Package ‘fslr’

February 2, 2018

Type Package

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

Version 2.17.3

Maintainer John Muschelli <muschelli2@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, matrixStats, R.utils, graphics, grDevices, stats, utils

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

License GPL-3

VignetteBuilder knitr

Suggests knitr, rmarkdown, covr

BugReports https://github.com/muschelli2/fslr/issues

SystemRequirements FSL

RoxygenNote 6.0.1.9000

Encoding UTF-8

NeedsCompilation no

Author John Muschelli [aut, cre]

Repository CRAN

Date/Publication 2018-02-02 16:45:53 UTC

R topics documented:

aux.file-methods .......................................................... 6
bitpix-methods ........................................................... 6
R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cal.max-methods</td>
<td>6</td>
</tr>
<tr>
<td>cal.min-methods</td>
<td>7</td>
</tr>
<tr>
<td>checkout</td>
<td>7</td>
</tr>
<tr>
<td>check_file</td>
<td>8</td>
</tr>
<tr>
<td>datatype-methods</td>
<td>8</td>
</tr>
<tr>
<td>data_type-methods</td>
<td>9</td>
</tr>
<tr>
<td>descrip-methods</td>
<td>9</td>
</tr>
<tr>
<td>dim._-methods</td>
<td>9</td>
</tr>
<tr>
<td>download_fsl</td>
<td>10</td>
</tr>
<tr>
<td>dtifit</td>
<td>10</td>
</tr>
<tr>
<td>eddy</td>
<td>11</td>
</tr>
<tr>
<td>eddy_correct</td>
<td>12</td>
</tr>
<tr>
<td>fast</td>
<td>13</td>
</tr>
<tr>
<td>fast.help</td>
<td>14</td>
</tr>
<tr>
<td>flirt</td>
<td>14</td>
</tr>
<tr>
<td>flirt.help</td>
<td>15</td>
</tr>
<tr>
<td>flirt_apply</td>
<td>15</td>
</tr>
<tr>
<td>fnirt</td>
<td>16</td>
</tr>
<tr>
<td>fnirt.help</td>
<td>17</td>
</tr>
<tr>
<td>fnirt_with_affine</td>
<td>17</td>
</tr>
<tr>
<td>fnirt_with_affine_apply</td>
<td>18</td>
</tr>
<tr>
<td>fslabs.help</td>
<td>19</td>
</tr>
<tr>
<td>fslacos.help</td>
<td>19</td>
</tr>
<tr>
<td>fsladd.help</td>
<td>20</td>
</tr>
<tr>
<td>fsland</td>
<td>20</td>
</tr>
<tr>
<td>fslasin.help</td>
<td>21</td>
</tr>
<tr>
<td>fslatan.help</td>
<td>21</td>
</tr>
<tr>
<td>fslbet.help</td>
<td>22</td>
</tr>
<tr>
<td>fslbin.help</td>
<td>23</td>
</tr>
<tr>
<td>fslbinv.help</td>
<td>23</td>
</tr>
<tr>
<td>fslchfiletype</td>
<td>24</td>
</tr>
<tr>
<td>fslchfiletype.help</td>
<td>24</td>
</tr>
<tr>
<td>fslcmd</td>
<td>25</td>
</tr>
<tr>
<td>fslcog</td>
<td>26</td>
</tr>
<tr>
<td>fslcos.help</td>
<td>27</td>
</tr>
<tr>
<td>fslcpgeom</td>
<td>27</td>
</tr>
<tr>
<td>fslcpgeom.help</td>
<td>28</td>
</tr>
<tr>
<td>fsldir</td>
<td>28</td>
</tr>
<tr>
<td>fsldiv.help</td>
<td>29</td>
</tr>
<tr>
<td>fsledge.help</td>
<td>29</td>
</tr>
<tr>
<td>fslenentropy</td>
<td>30</td>
</tr>
<tr>
<td>fslepi_reg</td>
<td>30</td>
</tr>
<tr>
<td>fslrode.help</td>
<td>31</td>
</tr>
<tr>
<td>fslexp.help</td>
<td>32</td>
</tr>
<tr>
<td>fsliill.help</td>
<td>32</td>
</tr>
<tr>
<td>fsliill2</td>
<td>33</td>
</tr>
<tr>
<td>fslgetorient</td>
<td>34</td>
</tr>
<tr>
<td>fslhd</td>
<td>34</td>
</tr>
</tbody>
</table>
R topics documented:

fslhd.help ................................................................. 35
fslhd.parse ................................................................. 35
fslhelp ................................................................. 36
fslindex.help ............................................................. 36
fsllog.help ................................................................. 37
fslmask.help ................................................................. 38
fslmaths.help ................................................................. 38
fslmax ................................................................. 39
fslmean ................................................................. 39
fslmerge.help ................................................................. 40
fslmul.help ................................................................. 40
fslnan.help ................................................................. 41
fslnanm.help ................................................................. 41
fslor ................................................................. 42
fslorient ................................................................. 43
fslorient.help ............................................................... 43
fslorienter ................................................................. 44
fslrand.help ................................................................. 44
fslrandn.help ................................................................. 45
fslrange ................................................................. 45
fslrecip.help ................................................................. 46
fslrem.help ................................................................. 47
fslreorient2std ............................................................... 47
fslreorient2std.help ........................................................... 48
fslrobustfov ................................................................. 48
fslrobustfov.help ............................................................. 49
fslroi ................................................................. 49
fslsd ................................................................. 50
fslsin ................................................................. 51
fslsin.help ................................................................. 52
fslslicer.timer ................................................................. 52
fslsmooth.help ................................................................. 53
fslsmooth_in_mask ............................................................ 54
fslsplit ................................................................. 55
fslsplit.help ................................................................. 56
fslsqr.....help ................................................................. 56
fslsqrt.help ................................................................. 57
fslstats ................................................................. 57
fslstats.help ................................................................. 58
fslsub.help ................................................................. 58
fslsub2.help ................................................................. 59
fslsum ................................................................. 60
fslswapdim.help ............................................................. 60
fsltan.help ................................................................. 61
fslthresh.help ................................................................. 61
fslval ................................................................. 62
fslval.help ................................................................. 62
fslview ................................................................. 63
R topics documented:

- fslview.help ........................................... 63
- fslvol .................................................. 64
- fslvolume .............................................. 64
- fslxor ................................................... 65
- fsl_abs .................................................. 66
- fsl_acos ................................................ 66
- fsl_add .................................................. 67
- fsl_anat ................................................. 68
- fsl_anat.help ......................................... 69
- fsl_applywarp ......................................... 69
- fsl_applywarp.help .................................... 70
- fsl_asin ................................................ 70
- fsl_atan ................................................ 71
- fsl_atlas_dir .......................................... 72
- fsl_avscale .......................................... 72
- fsl_bet .................................................. 73
- fsl_biasmcorrect ...................................... 74
- fsl_bin .................................................. 74
- fsl_binv ............................................... 75
- fsl_bin_tab ............................................ 76
- fsl_cos .................................................. 77
- fsl_data_dir .......................................... 77
- fsl_dice ............................................... 78
- fsl_dilate .............................................. 78
- fsl_div .................................................. 79
- fsl_edge ............................................... 80
- fsl_erode .............................................. 81
- fsl_exp .................................................. 82
- fsl_fill ................................................ 83
- fsl_index ............................................... 84
- fsl_log .................................................. 85
- fsl_mask ............................................... 85
- fsl_maths .............................................. 86
- fsl_merge .............................................. 87
- fsl_mul .................................................. 88
- fsl_nan ................................................. 89
- fsl_nanm ............................................... 89
- fsl_rand ............................................... 90
- fsl_randn ............................................. 91
- fsl_recip .............................................. 92
- fsl_rem ................................................ 93
- fsl_smooth .......................................... 94
- fsl_sqr ................................................ 95
- fsl_sqrt ............................................... 96
- fsl_std_dir .......................................... 96
- fsl_sub ............................................... 97
- fsl_sub2 .............................................. 98
- fsl_swapdim ......................................... 99
topics documented:

- fsl_tan ............................................................. 100
- fsl_thresh ......................................................... 100
- fsl_version ......................................................... 102
- get.fsl ............................................................. 102
- get.fsloutput ...................................................... 103
- get.imgext ......................................................... 103
- getForms .......................................................... 104
- have.fsl ............................................................ 104
- intent_code-methods ............................................. 105
- intent_name-methods ............................................. 105
- intent_p1-methods ............................................... 106
- intent_p2-methods ............................................... 106
- intent_p3-methods ............................................... 106
- magic-methods .................................................... 107
- mcflirt ............................................................. 107
- mcflirt.help ...................................................... 108
- melodic ............................................................ 108
- melodic.help ...................................................... 109
- mid_sagittal_align .............................................. 109
- mni_fname ........................................................ 110
- mni_img ........................................................... 110
- parse_avscale .................................................... 111
- pixdim-methods ................................................... 111
- probtrackx ......................................................... 112
- qform.character-method ........................................ 114
- qform_code-methods ............................................. 114
- readrpi ........................................................... 115
- reverse_rpi_orient .............................................. 115
- rpi_orient ........................................................ 116
- run_first_all ..................................................... 116
- run_first_all.help .............................................. 117
- scl_inter-methods ............................................... 118
- scl_slope-methods ............................................... 118
- sform_code-methods ............................................. 118
- sizeof_hdr-methods ............................................. 119
- slice_code-methods ............................................. 119
- slice_duration-methods ........................................ 120
- slice_end-methods .............................................. 120
- slice_start-methods ............................................ 121
- susan ............................................................. 121
- susan.help ....................................................... 122
- toffset-methods ................................................ 123
- topup ............................................................. 123
- vox_offset-methods ............................................. 124
- xfibres ............................................................ 125

Index .............................................................. 127
aux.file-methods  

Extract Image aux.file attribute

**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  

Extract Image bitpix attribute

**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  

Extract Image cal.max attribute

**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`
cal.min-methods

Extract Image cal.min attribute

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

checkout

Determine of Q and S forms are consistent

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",
                     "MNIST Tie2mm.nii.gz")
  forms = getForms(mnifile)
  checkout(forms)
}
```
check_file  

Wrapper for `getForms` with filename

Description

Checking the q/s-forms for a header

Usage

```r
check_file(file, ...)
```

Arguments

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

Value

result of `checkout`

Examples

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

datatype-methods  

Extract Image datatype attribute

Description

datatype method for character types

Usage

```r
## 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

**Description**

Download FSL Tarball

**Usage**

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**

dtifit(infile, bvecs, bvals, mask = NULL, outprefix = NULL, opts = "", bet.opts = "", verbose = TRUE, sse = FALSE, save_tensor = FALSE, grad_image = NULL)

**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
- prefix_V1 - 1st eigenvector
- prefix_V2 - 2nd eigenvector
- prefix_V3 - 3rd eigenvector
- prefix_L1 - 1st eigenvalue
- prefix_L2 - 2nd eigenvalue
- prefix_L3 - 3rd eigenvalue
- prefix_MD - mean diffusivity
- prefix_FA - fractional anisotropy
- prefix_MO - mode of the anisotropy (oblate ~ -1; isotropic ~ 0; prolate ~ 1)
- prefix_S0 - raw T2 signal with no diffusion weighting
optional output
If sse = True, then the additional file will be present: prefix_sse - Sum of squared error
If save_tensor = True, then the additional file will be present: prefix_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, outfile = NULL,
retimg = TRUE, opts = "", verbose = TRUE, ...)

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
The format of this file is identical to that used by topup (though the parameter is called --data in there).
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.

**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

...  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**

`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.
**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 = "", out_type = c("seg",
     "mixeltype", "pve_0", "pve_1", "pve_2", "pveseg"), verbose = TRUE,...)

fsl_fast(..., outfile = tempfile(fileext = "niigz"L retimg = falseL

fslfast(...)
```

**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
- `out_type` (character) Suffix to grab from outfile. For example, output filename is paste0(outfile, ",", out_type)  
- `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**

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

*FAST help*

**Description**

This function calls fast’s help

**Usage**

```r
fast.help()
```

**Value**

Prints help output and returns output as character vector

**Examples**

```r
if (haveNfsl()) {
  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**

```r
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`.
**flirt.help**

```
intern (logical) pass to system
opts (character) additional options to FLIRT
verbose (logical) print out command before running
...
```

**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**

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

---

**flirt_apply**

*Apply Warp from FLIRT*

**Description**

This function applies a matrix from flirt to other images

**Usage**

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

Register using FNIRT

Arguments

infile (character) input filename
reffile (character) reference image to be registered to
initmat (character) Matrix of transformation
outfile (character) output filename
retime (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 fnirt to register infile to reffile and either saves the image or returns an object of class nifti

Usage

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

Arguments

infile (character) input filename
reffile (character) reference image to be registered to
outfile (character) output filename
retime (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
**fnirt.help**

---

### Description

This function calls fnirt’s help

### Usage

```r
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

```r
fnirt_with_affine(infile, reffile, flirt.omat = NULL, flirt.outfile = NULL,
                  outfile = NULL, retimg = TRUE, roorient = 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
- `roorient` (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`
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
fslabs.help

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**  
*fsladd Help*

**Description**

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

**Usage**

`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**

`fsland(fileL, fileR, ...)`

`fsl_and(fileL, fileR, ...)`

**Arguments**

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

```
fslasin.help
```

---

**Description**

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

**Usage**

```
fslasin.help()
```

**Arguments**

```
...
```

passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector

**Examples**

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

---

**Description**

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

**Usage**

```
fslatan.help()
```
Arguments

... passed to fslmaths.help

Value

Prints help output and returns output as character vector

Examples

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

flsbet.help  Help for FSL BET

Description

This function calls bet’s help

Usage

flsbet.help(betcmd = c("bet2", "bet"))

Arguments

betcmd  (character) Get help for bet or bet2 function

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  flsbet.help()
  flsbet.help("bet")
}
fslbin.help

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()  
}

fslbinv.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()  
}
fslchfiletype

_FSL Change file type_

**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` 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) (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

_by fslchfiletype help_

**Description**

This function calls `fslchfiletype`'s help

**Usage**

```r
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, ...)
```

**Arguments**

- `func` (character) FSL function
- `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 `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
- `no outfile` (logical) is there an output file in the arguments of the FSL function?
- `...` 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.
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

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**

```r
fslcos.help(...)  
```

**Arguments**

... passed to fslmaths.help

**Value**

Prints help output and returns output as character vector

**Examples**

```r
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.

---

### fslcpgeom.help

<table>
<thead>
<tr>
<th>Usage</th>
<th>fslcpgeom.help()</th>
</tr>
</thead>
</table>

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

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
</table>
| if (have.fsl()){
  fslcpgeom.help()
} |

---

### fsldir

<table>
<thead>
<tr>
<th>Usage</th>
<th>fsldir()</th>
</tr>
</thead>
</table>

| Value | Character path |

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

**fsldiv Help**

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

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

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

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

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

fsledge.help

**fsledge Help**

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

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

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

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

**Examples**
```r
if (have.fsl){
  fsledge.help()
}
```
### fslentropy

**Image Mean Entropy**

**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

**Register EPI images to Structural image**

**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 = "", ...) fslepi_reg(..., outfile = tempfile(fileext = "nii.gz"), retimg = FALSE) epi_reg(..., outfile = tempfile(fileext = "nii.gz"), retimg = FALSE)
```
**Arguments**

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

**Value**

If `reting` 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`'s help, as `fslerode` is a wrapper for `fslmaths`.

**Usage**

`fslerode.help(...)`

**Arguments**

... passed to `fslmaths.help`
Value

Prints help output and returns output as character vector

Examples

```r
if (have.fsl()){
  fslrero.de.help()
}
```

---

### fslexp

**Description**

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

**Usage**

`fslexp.help(...)`

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector

**Examples**

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

---

### fslfill

**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

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

---

**fslfill2**  
*Fill image holes with dilation then erosion*

---

**Description**

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

**Usage**

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

**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.
- `reimg` (logical) return image of class nifti
- `reorient` (logical) If reimg, 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.
**FSL Orientation Wrappers**

### fslgetorient

**Description**

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

**Usage**

```r
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**

```r
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

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

---

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

Description

This function takes in a FSL header and parses the components

Usage

```r
fslhd.parse(hd)
```

Arguments

- `hd` (character) header from `fslhd`
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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>func_name</td>
<td>FSL function name</td>
</tr>
<tr>
<td>help.arg</td>
<td>Argument to print help, usually &quot;--help&quot;</td>
</tr>
<tr>
<td>extra.args</td>
<td>Extra arguments to be passed other than --help</td>
</tr>
</tbody>
</table>

Value

Prints help output and returns output as character vector

fslindex

fslindex Help

Description

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

Usage

fslindex.help(...)
### fslllog.help

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector

**Examples**

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

---

**Description**

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

**Usage**

`fslllog.help(...)`

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector

**Examples**

```r
if (have.fsl()){
  fslllog.help()
}
```
**fslmask.help**

**fslmask Help**

**Description**

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

**Usage**

```r
fslmask.help(...)  
```

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector.

**Examples**

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

---

**fslmaths.help**

**FSL Maths Help**

**Description**

This function calls fslmaths's help.

**Usage**

```r
fslmaths.help()  
```

**Value**

Prints help output and returns output as character vector.

**Examples**

```r
if (have.fsl()){
  fslmaths.help()  
}
```
fslmax  

*Get min/max of an image*

**Description**

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

**Usage**

```r
fslmax(file, ...)  
fslmin(file, ...)
```

**Arguments**

- `file` (character) filename of image to be checked
- `NNN` options passed to `fslrange`

**Value**

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

**Examples**

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

---

fslmean  

*Image Mean*

**Description**

Estimates Mean of Image from FSL.

**Usage**

```r
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

Description

This function calls fslmerge’s help

Usage

def fslmergeHelp()

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
    fslmergeHelp()
}

fslmul Help

Description

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

Usage

def fslmulHelp(...)

Arguments

... passed to fslmathsHelp

Value

Prints help output and returns output as character vector
Examples

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

Description

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

Usage

```
fslnan.help(...)```

Arguments

```
... passed to fslmaths.help```

Value

Prints help output and returns output as character vector.

Examples

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

Description

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

Usage

```
fslnanm.help(...)```

Arguments

```
... passed to fslmaths.help```
fslor

**Value**

Prints help output and returns output as character vector.

**Examples**

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

---

**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, inter = FALSE, ...

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  

**fslorient**  

**FSL Orient**

**Description**

This function calls `fslorient`

**Usage**

```r
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**

```r
fslorient.help()
```

**Value**

Prints help output and returns output as character vector

**Examples**

```r
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**

```r
fslrandn.help(...) 
```

**Arguments**

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

**Value**

Prints help output and returns output as character vector.

**Examples**

```r
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**

```r
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

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

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

fslreorient2std

FSL Orient to MNI

Description

This function calls fslreorient2std

Usage

fslreorient2std(file, retimg = TRUE, reorient = FALSE, intern = FALSE, 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
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.

---

**fslreorient2std.help**  
*fslreorient2std help*

Description

This function calls `fslreorient2std`'s help

Usage

```r
fslreorient2std.help()
```

Value

Prints help output and returns output as character vector

Examples

```r
if (have.fsl()){
  fslreorient2std.help()
}
```

---

**fslrobustfov**  
*FSL Robust Field of View*

Description

This function calls `robustfov` to automatically crop the image

Usage

```r
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.

---

**fslrobustfov.help**  
*FSL Robust Field of View Help*

---

**Description**

This function calls `robustfov help`

**Usage**

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

---

**fslroi**  
*FSL ROI*

---

**Description**

This function calls `fslroi`

**Usage**

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

`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
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

**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, ...)```

```
fsl_slicetimer(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)```
Arguments

file (character) image to be manipulated
outfile (character) resultant image name (optional)
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

Description

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

Usage

fslsmooth.help(...)
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 = 1L, mask = NULL, ...)
```

**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)
  )
}
```
fslsplit

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 (haveFsl()){
  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**
```r
... passed to fslmaths.help```

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

**Examples**
```r
if (haveFsl()){
  fslsqr.help()
}
```
fslsqrt.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

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(S$FLOAT32, cal.min = min(x),
                cal.max = max(x), pixdim = rep(1, TII}
    entropy = fslstats(img, opts='-E')
  })
```

---

**fslstats.help**

*FSL Stats Help*

**Description**

This function calls fslstats's help

**Usage**

```r
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**

```r
fslsub.help(...)```
Arguments

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

Value

Prints help output and returns output as character vector

Examples

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

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

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**
```
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**
```
if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard",
                     "MN152_T1_2mm.nii.gz")
  fslval(mnifile, keyword = "dim1")
}
```

---

**fslval.help**  
*fslval help*

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

**Usage**
```
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, ...)

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 checking

Value
character or logical depending on intern

Description
This function calls fslview's help

Usage
fslview.help()

Value
Prints help output and returns output as character vector
Examples

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

---

**fslvolume**

*FSL Volume in mL (or cubic centimeters)*

---

**Description**

This function wraps `fslsum` and `voxdim`

**Usage**

```r
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**

```r
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**

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

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

**Arguments**

- **file** (character) input image
- **file2** (character) image to be XOR’d
- **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
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)
reting (logical) return image of class nifti
file (character) input image to absolute value
reorient (logical) If reting, 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

Arc Cosine Transform Image using FSL

Description

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

Usage

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

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

Arguments

...
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

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

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

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

Value

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

Note

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

---

\textbf{fsl\_anat} \hspace{1cm} \textit{FSL Anatomical Processing Script}

**Description**

This function calls \texttt{fsl\_anat} from FSL

**Usage**

\begin{verbatim}
fsl\_anat(file, modality = c("T1", "T2", "PD"), outdir = NULL, 
         intern = FALSE, opts = "", verbose = TRUE, ...)
\end{verbatim}

**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 \texttt{system}
opts (character) operations to be passed to \texttt{fsl\_anat}
verbose (logical) print out command before running
... options passed to \texttt{checkimg}

**Value**

Result from system command, depends if \texttt{intern} is \texttt{TRUE} or \texttt{FALSE}. 

fsl_anat.help

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()
}

fsl_applywarp
Apply Warp from FNIRT

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

current or logical depending on intern

fsl_applywarp.help   FSL applywarp help

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
fsl_atan

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 = "NniiNgz"), retimg = FALSE)
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
```
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
fsl_biascorrect  

**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.

---

fsl_bin  

**Description**

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

**Usage**

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

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

Binarized Inverse Image using FSL

Description

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

Usage

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

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.

Note

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

---

### 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 vimage
- `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_cos**

*Cosine Transform Image using FSL*

**Description**

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

**Usage**

```r
fsl_cos(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
fslcos(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 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_data_dir**

*Get FSL's Data Directory*

**Description**

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

**Usage**

```r
fsl_data_dir()
```
fsl_dice

Calculate Dice Coefficient of 2 Binary images

Description

Creates a 2 by 2 table for

Usage

fsl_dice(x, y, ...)

Arguments

x: filename of logical or 0/1 image
y: filename of logical or 0/1 vimage
...: 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

fsl_dilate(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
fsl_dilate(file, outfile = NULL, retimg = TRUE, reorient = FALSE, intern = FALSE, kopts = "", opts = "", verbose = TRUE, ...)
Arguments

... additional arguments passed to \texttt{readnii}.

\texttt{outfile} (character) resultant dilated image name

\texttt{reorient} (logical) If \texttt{reorient}, should file be reoriented when read in? Passed to \texttt{readnii}.

\texttt{file} (character) image to be dilated

\texttt{intern} (logical) to be passed to \texttt{system}

\texttt{kopts} (character) options for kernel

\texttt{opts} (character) additional options to be passed to \texttt{fslmaths}

\texttt{verbose} (logical) print out command before running

Value

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

\begin{verbatim}
if (have.fsl()){
  system.time(
    dims = c(50, 50, 20)
    x = array(rnorm(prod(dims)), dim = dims)
    img = nifti(x, xdim = dims,
    datatype = convert.datatype($FLOAT32, cal.min = min(x),
    cal.max = max(x), xdim = rep(1, 4))
    mask = img > .5
    dilated = fsldilate(mask, kopts = "-kernel boxv 5", retimg=TRUE)
  }
}
\end{verbatim}

---

\texttt{fsl\_div} \hspace{1cm} \textit{Divide Images using FSL}

\textbf{Description}

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

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

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

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retrimg (logical) return image of class nifti
file (character) input image to estimate edge strength
reorient (logical) If retrimg, 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 retrimg 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

```r
fsl_erode(..., outfile = tempfile(fileext = ".nii.gz"), retrimg = FALSE)
fsl_erode(file, outfile = NULL, retrimg = TRUE, reorient = FALSE,
          intern = FALSE, kopts = ",", opts = ",", verbose = TRUE, ...)
```

Arguments

... additional arguments passed to readnii.
 outfile (character) resultant eroded image name
 retrimg (logical) return image of class nifti
 file (character) image to be eroded
 reorient (logical) If retrimg, 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 = fsl_erode(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)

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

**Arguments**

```
...  additional arguments passed to readnii.
outfile (character) resultant image name (optional)
reting (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)

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 > 0.5
        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)
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`
- `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_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

fsl_mask

Mask image using FSL

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

\[
fsl\_mask(...,\text{ reorient} = \text{FALSE}) \\
\]
\[
fslmask(\text{file}, \text{mask}, \text{outfile} = \text{NULL}, \text{reorient} = \text{FALSE}, \text{retimg} = \text{TRUE}, \text{intern} = \text{FALSE}, \text{opts} = "", \text{verbose} = \text{TRUE}, ...) \\
\]

Arguments

- \[\ldots\] additional arguments passed to \texttt{readnii}. 
- \texttt{outfile} (character) resultant masked image name 
- \texttt{retimg} (logical) return image of class nifti 
- \texttt{file} (character) image to be masked 
- \texttt{mask} (character) mask given for image 
- \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) additional options to be passed to \texttt{fslmask} 
- \texttt{verbose} (logical) print out command before running 

Value

Result from system command, depends if intern is TRUE or FALSE. if (have.fsl()) system.time( 
\[
x = \text{array(rnorm(1e6), dim = c(100, 100, 100))} \\
\text{img = nifti(x, dim= c(100, 100, 100), datatype = convert.datatype()$FLOAT32, cal.min = min(x), cal.max = max(x), pixdim = rep(1, 4))} \\
\text{mask = img > .5} \\
\text{masked = fslmask(img, mask = mask, retimg=TRUE)} \\
\]

Note

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

\[
\begin{array}{ll}
\text{fsl\_maths} & \text{FSL Maths} \\
\end{array}
\]

Description

This function calls \texttt{fslmaths} 

Usage

\[
fsl\_maths(...,\text{ outfile} = \text{tempfile(fileext = ".nii.gz"}, \text{retimg} = \text{FALSE}) \\
fslmaths(\text{file, outfile} = \text{NULL}, \text{retimg} = \text{TRUE}, \text{reorient} = \text{FALSE}, \text{intern} = \text{FALSE}, \text{opts} = "", \text{verbose} = \text{TRUE}, ...) \\
\]
fsl_merge

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
reoting (logical) return image of class nifti
file (character) image to be manipulated
reorient (logical) If reoting, 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

Value

If reoting 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 fslmerge to merge files on some dimension and either saves the image or returns an object of class nifti

Usage

fsl_merge(..., outfile = tempfile(fileext = "nii.gz"), reoting = FALSE)
fslmerge(infiles, direction = c("x", "y", "z", "t", "a"), outfile = NULL, reoting = TRUE, reorient = FALSE, intern = FALSE, verbose = TRUE, ...)
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
fsl_nan(...) outfile = tempfile(fileext = "\nii.gz"), retimg = FALSE)

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

fsl_nanm(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

_______________________________________

fsl_rand 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)

fsl_rand(file, outfile = NULL, retimg = TRUE, reorient = FALSE,
         intern = FALSE, opts = ",", ...)
**fsl_randn**

Add Random Standard Gaussian Noise Image using FSL

**Arguments**

... additional arguments passed to *readnii*.

outfile (character) resultant image name (optional)

reorient (logical) return image of class nifti

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

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

---

**Description**

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

**Usage**

```r
fsl_randn(..., outfile = tempfile(fileext = "nii.gz"), reorient = FALSE)
fslrandn(file, outfile = NULL, reorient = 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 add random standard to Gaussian noise

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 `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**

```r
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.
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
fsl_smooth

Gaussian smooth image using FSL

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
fsl_sqr  

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_sqr**  

*Square Image using FSL*

---

**Description**

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

**Usage**

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

fslsqr(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
- `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_sqrt

Square Root Image using FSL

Description

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

Usage

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

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()
Description

This function calls fslmaths -sub. The R function 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)
retimg (logical) return image of class nifti
file (character) input image
file2 (character) image to be subtracted
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_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
fsl_sub2(..., outfile = tempfile(fileext = "nii.gz"), retimg = FALSE)

fslsub2(file, outfile = NULL, retimg = TRUE, reorient = FALSE,
        intern = FALSE, verbose = TRUE, ...)

Arguments
... additional arguments passed to readnii.
outfile (character) name of resultant subsampled file
retimg (logical) return image of class nifti
file (character) filename of image to be subsampled
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
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))
    subsamp = fslsub2(img, retimg=TRUE)
    print(voxdim(subsamp))
  )
}
**fsl_swapdim**

**FSL Swap Dimensions**

**Description**

This function calls `fslswapdim`

**Usage**

```r
fsl_swapdim(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
fslswapdim(file, outfile = NULL, retimg = TRUE, reorient = FALSE,
           intern = FALSE, a = "x", b = "y", c = "z", verbose = 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

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

```r
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

```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()
                cal.min = min(x),
                cal.max = max(x), pixdim = rep(1, 4))
    thresh = fslthresh(img, thresh=0, uthresh = 2, retimg=TRUE)
  }
}
fsl_version

Find FSL Version

Description

Finds the FSL version from FSLSDIR/etc/fslversion

Usage

```r
fsl_version()
```

```r
fslversion()
```

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 FSLSDIR

Description

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

Usage

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

```r
get_fsl(add_bin = TRUE)
```

Arguments

| add_bin | Should bin be added to the fsl path? All executables are assumed to be in FSLSDIR/bin/. If not, and add_bin = FALSE, they will be assumed to be in FSLSDIR/. |
get.fsloutput

**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**  
_Determine FSL output type_

**Description**

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

**Usage**

```plaintext
get.fsloutput()
```

**Value**

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

---

**get.imgext**  
_Determine extension of image based on FSLOUTPUTTYPE_

**Description**

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

**Usage**

```plaintext
get.imgext()
```

**Value**

Extension for output type
getforms

*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 = file.path(fsldir(), "data", "standard",
                     "MNI152_T1_2mm.nii.gz")
  getforms(mnifile)
}
```

---

**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**

```r
have.fsl(...)  
```

```r
have_fsl(...)  
```

**Arguments**

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

Examples
have.fsl()
### intent_p1-methods

**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`

### intent_p2-methods

**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`

### intent_p3-methods

**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`
**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`

**Value**

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

\subsection*{Description}
This function calls \texttt{mcflirt}'s help

\subsection*{Usage}
\begin{verbatim}
mcflirt.help()
\end{verbatim}

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

\subsection*{Examples}
\begin{verbatim}
library(fslr)
if (have.fsl()){
  mcflirt.help()
}
\end{verbatim}

\section*{melodic  \textit{Run MELODIC ICA}}

\subsection*{Description}
This function calls \texttt{melodic}

\subsection*{Usage}
\begin{verbatim}
melodic(file, outdir = dirname(file), intern = FALSE, opts = ",",
        verbose = TRUE, ...)
\end{verbatim}

\subsection*{Arguments}
\begin{description}
\item[\texttt{file}] (character) image to be run
\item[\texttt{outdir}] (character) output directory. (Default \texttt{dirname(file)})
\item[\texttt{intern}] (logical) pass to \texttt{system}
\item[\texttt{opts}] (character) options for \texttt{melodic}
\item[\texttt{verbose}] (logical) print out command before running
\item[\ldots] arguments passed to \texttt{checkimg}
\end{description}

\subsection*{Value}
character or logical depending on \texttt{intern}
melodic.help

**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**

```
mid_sagittal_align(file, outfile = NULL, retimg = TRUE, opts = "", 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
- `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
mni_fname(mm = c("1", "0.5", "2"), brain = FALSE, linear = FALSE, mask = FALSE)

Arguments
- **mm**: Resolution (in mm) of the brain image (isotropic)
- **brain**: Should the brain be returned (default) or the T1 with the skull
- **linear**: Should the linearized MNI template be used
- **mask**: Should the mask be given? Generally, only MNI152_T1_1mm_brain_mask exists.

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

mni_img

Read MNI Filename

Description
Simple wrapper for reading in the MNI image constructed from mni_fname

Usage
mni_img(...)

Arguments
- ... Arguments passed to mni_fname

Value
Object of class nifti
parse_avscale

Description

This function parses the output from fsl_avscale into something more manageable

Usage

parse_avscale(av_out)

Arguments

av_out output from fsl_avscale, character vector

Value

List of output values

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 fslval
probtrackx

Probabilistic diffusion tractography with multiple fibre orientations

Description

This function wraps probtrackx2 from FSL.

Usage

probtrackx(samples = "merged", mask, seed, verbose = TRUE, out = NULL, dir = NULL, forcedir = FALSE, simple = NULL, network = FALSE, opd = NULL, pd = FALSE, fopd = NULL, os2t = FALSE, s2tastext = NULL, targetmasks = NULL, waypoints = NULL, waycond = c("AND", "OR"), wayorder = NULL, onewaycondition = FALSE, avoid = NULL, stop = NULL, omatrix1 = NULL, distthresh1 = NULL, omatrix2 = NULL, target2 = NULL, omatrix3 = NULL, target3 = NULL, ltarget3 = NULL, distthresh3 = 0, xfm = NULL, invxfm = NULL, seedref = NULL, meshspace = c("caret", "freesurfer", "first", "vox"), nsamples = 5000, nsteps = 2000, steplength = 0.5, distthresh = 0, cthr = 0.2, fibthresh = 0.01, loopcheck = FALSE, usef = FALSE, modeuler = FALSE, sampvox = 0, randfib = 0, fibst = 1, rseed = NULL, ...)

Arguments

samples  Basename for samples files - e.g. 'merged'
mask  Bet binary mask file in diffusion space
seed  Seed volume or list (ascii text file) of volumes and/or surfaces
verbose  Verbose level, [0-2]
out  Output file (default='fdt_paths')
dir  Directory to put the final volumes in - code makes this directory - default='logdir'
forcedir  Use the actual directory name given - i.e. don't add + to make a new directory
simple  Track from a list of voxels (seed must be a ASCII list of coordinates)
network  Activate network mode - only keep paths going through at least one of the other seed masks
opd  Output path distribution
pd  Correct path distribution for the length of the pathways
fopd  Other mask for binning tract distribution
os2t  Output seeds to targets
s2tastext  Output seed-to-target counts as a text file (default in simple mode)
targetmasks  File containing a list of target masks - for seeds_to_targets classification
waypoints  Waypoint mask or ascii list of waypoint masks - only keep paths going through ALL the masks
**waycond**  Waypoint condition. Either 'AND' (default) or 'OR'

**wayorder**  Reject streamlines that do not hit waypoints in given order. Only valid if waycond=AND

**onewaycondition**  
Apply waypoint conditions to each half tract separately

**avoid**  Reject pathways passing through locations given by this mask

**stop**  Stop tracking at locations given by this mask file

**omatrix1**  Output matrix1 - SeedToSeed Connectivity

**distthresh1**  Discards samples (in matrix1) shorter than this threshold (in mm - default=0)

**omatrix2**  Output matrix2 - SeedToLowResMask

**target2**  Low resolution binary brain mask for storing connectivity distribution in matrix2 mode

**omatrix3**  Output matrix3 (NxN connectivity matrix)

**target3**  Mask used for NxN connectivity matrix (or Nxn if lrtarget3 is set)

**lrtarget3**  Column-space mask used for NxN connectivity matrix

**distthresh3**  Discards samples (in matrix3) shorter than this threshold (in mm - default=0)

**xfm**  Transform taking seed space to DTI space (either FLIRT matrix or FNIRT warpfield) - default is identity

**invxfm**  Transform taking DTI space to seed space (compulsory when using a warpfield for seeds_to_dti)

**seedref**  Reference vol to define seed space in simple mode - diffusion space assumed if absent

**meshspace**  Mesh reference space - either 'caret' (default) or 'freesurfer' or 'first' or 'vox'

**nsamples**  Number of samples - default=5000

**nsteps**  Number of steps per sample - default=2000

**steplength**  Steplength in mm - default=0.5

**distthresh**  Discards samples shorter than this threshold (in mm - default=0)

**cthr**  Curvature threshold - default=0.2

**fibthresh**  Volume fraction before subsidiary fibre orientations are considered - default=0.01

**loopcheck**  Perform loopchecks on paths - slower, but allows lower curvature threshold

**usef**  Use anisotropy to constrain tracking

**modeuler**  Use modified euler streamlining

**sampvox**  Sample random points within x mm sphere seed voxels (e.g. –sampvox=5). Default=0

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

**rseed**  Random seed

...  Additional arguments
qform, character-method

*Extract NIfTI 3D Image Orientation*

**Description**

Gets q/s-forms from a character

**Usage**

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

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

**Arguments**

- `object` is a `nifti` object

---

qform_code-methods

*Extract Image qform_code attribute*

**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`

---

reverse_rpi_orient

**Reverse Reorientation an Image to RPI orientation**

**Description**

This function uses `fslswapdim` to reorient an image.

**Usage**

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

reverse_rpi_orient_file(file, 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)

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

run_first_all  
Run FIRST All

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

Usage
run_first_all(img, oprefix = tempfile(), brain_extracted = FALSE,
structures = NULL, affine = NULL, opts = "", verbose = TRUE)
Arguments

- **img**: Specifies the input image (T1-weighted)
- **oprefix**: Specifies the output image basename (extensions will be added to this)
- **brain_extracted**: Specifies that the input image has been brain extracted
- **structures**: A restricted set of structures to be segmented
- **affine**: Specifies the affine registration matrix to standard space (optional)
- **opts**: (character) Operations to be passed to `run_first_all`
- **verbose**: (logical) Print out command before running

Value

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

---

**Description**

This function calls `run_first_all`'s help

**Usage**

```r
run_first_all.help()
```

**Value**

Prints help output and returns output as character vector

**Examples**

```r
library(fslr)

if (have.fsl()){
  run_first_all.help()
}
```
**scl_inter-methods**

*Extract Image scl_inter attribute*

**Description**

scl_inter method for character types

**Usage**

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

**Arguments**

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

---

**scl_slope-methods**

*Extract Image scl_slope attribute*

**Description**

scl_slope method for character types

**Usage**

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

**Arguments**

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

---

**sform_code-methods**

*Extract Image sform_code attribute*

**Description**

sform_code method for character types

**Usage**

```r
## 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'
ssizeof_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**

```r
## 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**

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

**Arguments**

- **object**: is a filename to pass to fslval
slice_start-methods

Extract Image slice_start attribute

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).
susan.help

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


susan.help  FSL SUSAN Help

Description

This function calls susan’s help

Usage

susan.help()

Value

Prints help output and returns output as character vector

Examples

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


toffset-methods

Description

Gets toffset from a character

Usage

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

Arguments

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

---

topup

`topup` - calling FSL `topup`

Description

A tool for estimating and correcting susceptibility induced distortions

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, sqslambda = 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)
```

```r
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
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`
**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
xfibres

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

aux.file, character-method
  (aux.file-methods), 6
aux.file-methods, 6
avscale (fsl_avscale), 72

bedpostx (xfibres), 125
bitpix, character-method
  (bitpix-methods), 6
bitpix-methods, 6

cal.max, character-method
  (cal.max-methods), 6
cal.max-methods, 6
cal.min, character-method
  (cal.min-methods), 7
cal.min-methods, 7
check_file, 8
checking, 8, 34, 45, 57, 62, 63, 68, 104, 108
checkout, 7, 8

data_type, character-method
  (data_type-methods), 9
data_type-methods, 9
datatype, character-method
  (datatype-methods), 8
datatype-methods, 8
descrp, character-method
  (descrp-methods), 9
descrp-methods, 9
dim_, character-method (dim_-methods), 9
dim_-methods, 9
download.file, 10
download_fsl, 10
dtifit, 10

eddy, 11, 12
eddy_correct, 12
epi_reg (fslepi_reg), 30

fast, 13
fast.help, 14
flirt, 14, 15, 109
flirt.help, 15
flirt_apply, 15
flirt_apply.help (fsl.help), 15
fnirt, 16, 18, 69
fnirt.help, 17
fnirt_with_affine, 17
fnirt_with_affine_apply, 18
fsl_abs, 66
fsl_acos, 66
fsl_add, 67
fsl_anat, 68
fsl_anat.help, 69
fsl_and (fsland), 20
fsl_applywarp, 69
fsl_applywarp.help, 70
fsl_applywarp_apply, 71
fsl_atlas_dir, 72
fsl_avscale, 72, 111
fsl_bet, 73
fsl_bet_correct, 74
fsl_bin, 74
fsl_bin_tab, 76, 78
fsl_binv, 75
fsl_cos, 77
fsl_data_dir, 77
fsl_dice, 78
fsl_dilate, 78
fsl_dir (fsldir), 28
fsl_div, 79
fsl_edge, 80
fsl_epi_reg (fslepi_reg), 30
fsl_erode, 81
fsl_exp, 82
fsl_fast (fast), 13
fsl_fill, 83
fsl_index, 84
fsl_log, 85
fsl_mask, 85
fsl_maths, 86
fsl_merge, 87
fsl_mul, 88
fsl_nan, 89
fsl_nann, 89
fsl_or (fslor), 42
fsl_rand, 90
fsl_recip, 90
fsl_rem, 93
fsl_robustfov (fslrobustfov), 48
fsl_slicetimer (fslslicetimer), 52
fsl_smooth, 94
fsl_smooth_in_mask (fslsmooth_in_mask), 54
fsl_split (fslsplit), 55
fsl_sqr, 95
fsl_sqr (fsl_sqr), 95
fsl_sqt (fsl_std_dir), 96
fsl_sub, 97
fsl_sub2, 98
fsl_swapdim, 99
fsl_thre, 65, 100
fsl_topup (topup), 123
fsl_version, 102
fsl_xor (fslxor), 65
fslabs (fsl_abs), 66
fslabs.help, 19
fslacos (fsl_acos), 66
fslacos.help, 19
fsladd (fsl_add), 67
fsladd.help, 20
fsladd, 20
fslasin (fsl_asin), 70
fslasin.help, 21
fslatan (fsl_atan), 71
fslatan.help, 21
fslbet, 11, 125
fslbet (fsl_bet), 73
fslbet.help, 22
fslbin, 20, 42, 65
fslbin (fsl_bin), 74
fslbin.help, 23
fslinv (fsl_binv), 75
fslbinv.help, 23
fslchfiletype, 24
fslchfiletype.help, 24
fslcmd, 12, 25, 122
fslcog, 26
fslcos (fsl_cos), 77
fslcos.help, 27
fslcpgeom, 27
fslcpgeom.help, 28
fsldilate (fsl_dilate), 78
fsldir, 28
fsldiv (fsl_div), 79
fsldiv.help, 29
fsl_edge, 80
fsl_edge, 80
fslentropy, 30
fslentropy.help (fslstats.help), 58
fslepi_reg, 30
fsl_erode (fsl_erode), 81
fsl_erode.help, 31
fsl_exp (fsl_exp), 82
fsl_exp.help, 32
fat (fast), 13
fsl_fill, 33
fsl_fill (fsl_fill), 83
fsl_fill.help, 32
fsl_fill2, 33
fslgetorient, 34
fslgetqform (fslgetorient), 34
fslgetqform (fslgetorient), 34
fslgetqform (fslgetorient), 34
fslgetqform (fslgetorient), 34
fslhd, 34, 35, 104
fslhd.help, 35
fslhd.parse, 35
fslhelp, 36
fsl_index (fsl_index), 84
fsl_index.help, 36
fsl_log (fsl_log), 85
fsl_log, 85
fsl_log.help, 37
fslmask (fsl_mask), 85
fslmask.help, 38
fslmats (fsl_maths), 86
fslmats.help, 19–23, 27, 29, 31–33, 37, 38, 38, 40, 41, 44–47, 52, 53, 56, 57, 59, 61
fslmax, 39
fslmean, 39
fslmean_help (fslstats.help), 58
fslmerge (fsl_merge), 87
INDEX 129

fslmerge.help, 40
fslmin (fslmax), 39
fslmul, 20
fslmul (fsl_mul), 88
fslmul.help, 40
fslnan (fsl_nan), 89
fslnan.help, 41
fslnanm (fsl_nanm), 89
fslnanm.help, 41
fslor, 42
fslorient, 34, 43, 44
fslorient.help, 43
fslorint, 44
fslrand (fsl_rand), 90
fslrand.help, 44
fslrandn (fsl_randn), 91
fslrandn.help, 45
fslrange, 39, 45
fslrange.help (fslstats.help), 58
fslrecip (fsl_recip), 92
fslrecip.help, 46
fslrem (fsl_rem), 93
fslrem.help, 47
fslreorient2std, 47
fslreorient2std.help, 48
fslrobind, 48
fslrobind.help, 49
fslroi, 49
fslroi_time (fslroi), 49
fssl, 50
fsslslsd.help (fslstats.help), 58
fsslslsin, 51
fsslslsin.help, 52
fsslslslice, 52
fsslsmooth, 54
fsslsmooth (fsl_smooth), 94
fsslsmooth.help, 53
fsslsmooth_in_mask, 54
fsslsplit, 55
fsslsplit.help, 56
fsslqr (fsl_sqr), 95
fsslqr.help, 56
fsslqr (fsl_sqr), 96
fsslqr.help, 57
fsslstats, 30, 40, 51, 57, 60, 65
fsslstats.help, 58
fsslsub (fsl_sub), 97
fsslsub.help, 58
fsslsub2 (fsl_sub2), 98
fsslsub2.help, 59
fsslsum, 60, 64
fsslswapdim (fsl_swapdim), 99
fsslswapdim.help, 60
fssltau (fsl_tau), 100
fssltau.help, 61
fsslthresh (fsl_thresh), 100
fsslthresh.help, 61
fsslval, 6–9, 62, 105–107, 111, 114, 118–121, 123, 124
fsslval.help, 62
fsslversion (fsl_version), 102
fsslview, 63
fsslview.help, 63
fsslvol, 64
fsslvolume, 64
fsslxor, 65
get.fsl, 102, 104
get.fssloutput, 103
get.imgext, 103
get.fsl (get.fsl), 102
getForms, 7, 104
have.fsl, 104
have_fsl (have.fsl), 104

intent_code, character-method
(intent_code-methods), 105
intent_code-methods, 105
intent_name, character-method
(intent_name-methods), 105
intent_name-methods, 105
intent_p1, character-method
(intent_p1-methods), 106
intent_p1-methods, 106
intent_p2, character-method
(intent_p2-methods), 106
intent_p2-methods, 106
intent_p3, character-method
(intent_p3-methods), 106
intent_p3-methods, 106

magic, character-method (magic-methods), 107
magic-methods, 107
mcflirt, 107
mcflirt.help, 108
melodic, 108
melodic.help, 109
mid_sagittal_align, 109
mni_fname, 110, 110
mni_img, 110

nifti, 110

parse_avscale, 72, 111
pixdim, character-method
  (pixdim-methods), 111
pixdim-methods, 111
protrackx, 112

qform, character
  (qform,character-method), 114
qform, character-method, 114
qform_code, character-method
  (qform_code-methods), 114
qform_code-methods, 114

readnii, 13–18, 24, 25, 27, 28, 31, 33, 42, 43,
  47, 49–51, 53, 55, 65–71, 73–77,
  79–101, 107, 115, 121
readrpi, 115
reverse_rpi_orient, 115
reverse_rpi_orient_file
  (reverse_rpi_orient), 115
rpi_orient, 115, 116
rpi_orient_file, 115
rpi_orient_file(rpi_orient), 116
run_first_all, 116
run_first_all.help, 117

scl_inter, character-method
  (scl_inter-methods), 118
scl_inter-methods, 118
scl_slope, character-method
  (scl_slope-methods), 118
scl_slope-methods, 118
scl_inter, character-method
  (scl_inter-methods), 118
scl_slope, character-method
  (scl_slope-methods), 118
scl_slope-methods, 118
scl_inter, character-method
  (scl_inter-methods), 118
scl_slope, character-method
  (scl_slope-methods), 118
scl_slope-methods, 118
sizeof_hdr, character-method
  (sizeof_hdr-methods), 119
slice_code, character-method
  (slice_code-methods), 119
slice_code-methods, 119
slice_duration, character-method
  (slice_duration-methods), 120
slice_duration-methods, 120
slice_end, character-method
  (slice_end-methods), 120
slice_end-methods, 120
slice_start, character-method
  (slice_start-methods), 121
slice_start-methods, 121
susan, 121
susan.help, 122
system, 13, 15–18, 24, 25, 27, 31, 33, 42, 43,
  47, 49–51, 53, 63, 65–71, 73–77,
  79–101, 107, 108, 117, 121, 126
toffset, character-method
  (toffset-methods), 123
toffset-methods, 123
topup, 123

vox_offset, character-method
  (vox_offset-methods), 124
vox_offset-methods, 124
voxdim, 64

xfibres, 125