Package `fslr`

March 7, 2017

**Type** Package

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

**Version** 2.12

**Author** John Muschelli <muschelli2@gmail.com>

**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

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

**SystemRequirements** FSL <http://fsl.fmrib.ox.ac.uk/fsl/fslwiki/>

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2017-03-07 01:06:25

**R topics documented:**

- aux.file-methods .......................................................... 6
- bitpix-methods .................................................................. 6
- cal.max-methods ................................................................. 6
cal.min-methods ................................................. 7
checkout ......................................................... 7
check_file ....................................................... 8
datatype-methods ................................................. 8
data_type-methods .............................................. 9
descr-methods .................................................. 9
dim._-methods ................................................ 9
download_fsl ................................................... 10
dtifit ............................................................. 10
eddy_correct ................................................... 11
fast .............................................................. 12
fast.help .......................................................... 12
flirt .............................................................. 13
flirt.help .......................................................... 14
flirt.apply ....................................................... 14
fnirt .............................................................. 15
fnirt.help .......................................................... 15
fnirt_with_affine .............................................. 16
fnirt_with_affine_apply ....................................... 17
fslabs.help ..................................................... 18
fslacos.help .................................................... 18
fsladd.help ...................................................... 19
fslasin.help ...................................................... 19
fslatan.help ..................................................... 20
fslbet.help ....................................................... 20
fslbin.help ...................................................... 21
fslbinv.help ..................................................... 21
fslchfiletype ................................................... 22
fslchfiletype.help ............................................. 23
fslcmd .......................................................... 23
fslcog .......................................................... 24
fslcos.help .................................................... 25
fslcpgeom ....................................................... 25
fslcpgeom.help .................................................. 26
fsldiv.help ...................................................... 26
fsledge.help ..................................................... 27
fslentropy ....................................................... 28
fslerode.help ................................................... 28
fslexp.help ..................................................... 29
fslfill.help ..................................................... 29
fslfill2 .......................................................... 30
fslgetorient ..................................................... 31
fslhd ............................................................ 31
fslhd.help ....................................................... 32
fslhd.parse ..................................................... 32
fshelp ........................................................... 33
fslindex.help ................................................... 33
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>fsllog.help</td>
<td>34</td>
</tr>
<tr>
<td>fslmask.help</td>
<td>35</td>
</tr>
<tr>
<td>fslmaths.help</td>
<td>35</td>
</tr>
<tr>
<td>fslmax</td>
<td>36</td>
</tr>
<tr>
<td>fslmean</td>
<td>36</td>
</tr>
<tr>
<td>fslmerge.help</td>
<td>37</td>
</tr>
<tr>
<td>fslmul.help</td>
<td>37</td>
</tr>
<tr>
<td>fslnan.help</td>
<td>38</td>
</tr>
<tr>
<td>fslnamn.help</td>
<td>38</td>
</tr>
<tr>
<td>fslorient</td>
<td>39</td>
</tr>
<tr>
<td>fslorient.help</td>
<td>40</td>
</tr>
<tr>
<td>fslorienter</td>
<td>40</td>
</tr>
<tr>
<td>fslrand.help</td>
<td>41</td>
</tr>
<tr>
<td>fslrandn.help</td>
<td>41</td>
</tr>
<tr>
<td>fslrange</td>
<td>42</td>
</tr>
<tr>
<td>fslrecip.help</td>
<td>42</td>
</tr>
<tr>
<td>fslrem.help</td>
<td>43</td>
</tr>
<tr>
<td>fslreorient2std</td>
<td>43</td>
</tr>
<tr>
<td>fslreorient2std.help</td>
<td>44</td>
</tr>
<tr>
<td>fslrobustfov</td>
<td>45</td>
</tr>
<tr>
<td>fslrobustfov.help</td>
<td>45</td>
</tr>
<tr>
<td>fslroi</td>
<td>46</td>
</tr>
<tr>
<td>fslsd</td>
<td>47</td>
</tr>
<tr>
<td>fslsin</td>
<td>47</td>
</tr>
<tr>
<td>fslsin.help</td>
<td>48</td>
</tr>
<tr>
<td>fsl slicetimer</td>
<td>48</td>
</tr>
<tr>
<td>fslsmooth.help</td>
<td>49</td>
</tr>
<tr>
<td>fslsmooth_in_mask</td>
<td>50</td>
</tr>
<tr>
<td>fslsplit</td>
<td>51</td>
</tr>
<tr>
<td>fslsplit.help</td>
<td>52</td>
</tr>
<tr>
<td>fslsqr.help</td>
<td>52</td>
</tr>
<tr>
<td>fslsqrn.help</td>
<td>53</td>
</tr>
<tr>
<td>fslstats</td>
<td>53</td>
</tr>
<tr>
<td>fslstats.help</td>
<td>54</td>
</tr>
<tr>
<td>fslsub.help</td>
<td>54</td>
</tr>
<tr>
<td>fslsub2a2b2hlp</td>
<td>55</td>
</tr>
<tr>
<td>fslsum</td>
<td>56</td>
</tr>
<tr>
<td>fslswapdim.help</td>
<td>56</td>
</tr>
<tr>
<td>fsltan.help</td>
<td>57</td>
</tr>
<tr>
<td>fslthresh.help</td>
<td>57</td>
</tr>
<tr>
<td>fslval</td>
<td>58</td>
</tr>
<tr>
<td>fslvalnhelp</td>
<td>58</td>
</tr>
<tr>
<td>fslview</td>
<td>59</td>
</tr>
<tr>
<td>fslviewnhelp</td>
<td>59</td>
</tr>
<tr>
<td>fslvol</td>
<td>60</td>
</tr>
<tr>
<td>fslvolume</td>
<td>60</td>
</tr>
<tr>
<td>fsl_abs</td>
<td>61</td>
</tr>
<tr>
<td>fsl_acos</td>
<td>62</td>
</tr>
</tbody>
</table>
### R topics documented:

- `fsl_add` ......................................................... 62
- `fsl_anat` ......................................................... 63
- `fsl_anat.help` .................................................. 64
- `fsl_applywarp` ............................................... 65
- `fsl_applywarp.help` .......................................... 65
- `fsl_asin` ......................................................... 66
- `fsl_atan` ......................................................... 66
- `fsl_atlas_dir` .................................................. 67
- `fsl_avscale` ..................................................... 68
- `fsl_bet` ......................................................... 68
- `fsl_biascorrect` ............................................... 69
- `fsl_bin` ......................................................... 70
- `fsl_binv` ......................................................... 71
- `fsl_bin_tab` ..................................................... 72
- `fsl_cos` ......................................................... 72
- `fsl_data_dir` .................................................... 73
- `fsl_dice` ......................................................... 74
- `fsl_dilate` ....................................................... 74
- `fsl_div` ......................................................... 75
- `fsl_edge` ......................................................... 76
- `fsl_erode` ....................................................... 77
- `fsl_exp` ......................................................... 78
- `fsl_fill` ......................................................... 79
- `fsl_index` ....................................................... 80
- `fsl_log` ......................................................... 81
- `fsl_mask` ......................................................... 81
- `fsl_maths` ....................................................... 82
- `fsl_merge` ....................................................... 83
- `fsl_mul` ......................................................... 84
- `fsl_nan` ......................................................... 85
- `fsl_nanm` ....................................................... 85
- `fsl_rand` ....................................................... 86
- `fsl_randn` ...................................................... 87
- `fsl_recip` ....................................................... 88
- `fsl_rem` ......................................................... 89
- `fsl_smooth` ..................................................... 90
- `fsl_sqr` ......................................................... 91
- `fsl_sqrt` ......................................................... 92
- `fsl_std_dir` .................................................... 92
- `fsl_sub` ......................................................... 93
- `fsl_sub2` ....................................................... 94
- `fsl_swapdim` ................................................... 95
- `fsl_tan` ......................................................... 96
- `fsl_thresh` ..................................................... 96
- `fsl_version` .................................................... 98
- `get.fsl` ......................................................... 98
- `get.fsloutput` .................................................. 99
- `get.imgext` ..................................................... 99
R topics documented:

getForms ................................................................. 100
have.fsl ................................................................. 100
intent_code-methods ................................................. 101
intent_name-methods .................................................. 101
intent_p1-methods ..................................................... 102
intent_p2-methods ..................................................... 102
intent_p3-methods ..................................................... 102
magic-methods ......................................................... 103
mcflirt ................................................................. 103
mcflirt.help ............................................................ 104
melodic ................................................................. 104
melodic.help ............................................................ 105
mid_sagittal_align ...................................................... 105
mni_fname .............................................................. 106
mni_img ................................................................. 106
parse_avscale ............................................................ 107
pixdim-methods .......................................................... 107
protrackx ............................................................... 108
qform.character-method ............................................ 110
qform_code-methods .................................................. 110
readrpi ................................................................. 111
reverse_rpi_orient ...................................................... 111
rpi_orient .............................................................. 112
run_first_all ............................................................ 112
run_first_all.help ...................................................... 113
scl_inter-methods ...................................................... 114
scl_slope-methods ...................................................... 114
sform_code-methods ................................................... 114
sizeof_hdr-methods .................................................... 115
slice_code-methods ..................................................... 115
slice_duration-methods .............................................. 116
slice_end-methods ...................................................... 116
slice_start-methods .................................................... 117
susan ................................................................. 117
susan.help ............................................................. 118
toffset-methods ......................................................... 119
topup ................................................................. 119
vox_offset-methods ................................................... 120
xfibres ................................................................. 121

Index 123
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

Description

cal_min method for character types

Usage

## 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

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",  
  "MN152_T1_2mm.nii.gz")
  forms = getForms(mnifile)
  checkout(forms)
}
```
check_file  
Wrapper for `getForms` with filename

**Description**

Checking the q/s-forms for a header

**Usage**

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

**Arguments**

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

**Value**

result of `checkout`

**Examples**

```r
if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard",
                      "MN152_T1_2mm.nii.gz")
  check_file(mnifile)
}
```

---

extractImage.datatype

**Description**

datatype method for character types

**Usage**

```r
# S4 method for signature 'character'
extractImage.datatype(object)
```

**Arguments**

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

---

Extract Image data_type attribute

**Description**

data_type method for character types

**Usage**

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

**Arguments**

- `object` is a filename to pass to fslval

---

descrip-methods

---

Extract Image descrip attribute

**Description**

descrip method for character types

**Usage**

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

**Arguments**

- `object` is a filename to pass to fslval

---

dim_-methods

---

Extract Image dim_ attribute

**Description**

dim_ method for class character

**Usage**

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

**Arguments**

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

**Description**
Download FSL Tarball

**Usage**

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)

**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
eddy_correct

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

Value
Vector of character filenames of output. See Note

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

eddy_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, retimg = TRUE, reorient = FALSE, intern = FALSE, opts = "", out_type = c("seg", "mixeltype", "pve_0", "pve_1", "pve_2", "pveseg"), 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 `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
- **...** (logical): 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.

---

**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 (have.fsl()){
  fast.help()
}
```

---

**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`.
- `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`
**flirt.help**

**FLIRT help**

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

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

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

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

**flirt_apply**

**Apply Warp from FLIRT**

**Description**
This function applies a matrix from flirt to other images

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

**Arguments**
- `infile` (character) input filename
- `reffile` (character) reference image to be registered to
- `initmat` (character) Matrix of transformation
- `outfile` (character) output filename
- `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.
fnirt

Value

character or logical depending on intern

fnirt Register using FNIRT

Description

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

Usage

fnirt(infileL reffileL outfile = NULLL, retimg = TRUEL, reorient = FALSEL,
intern = FALSEL, opts = "", verbose = TRUEL, ...)

Arguments

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

Value

character or logical depending on intern

fnirt.help FNIRT help

Description

This function calls fnirt’s help

Usage

fnirt.help()

Value

Prints help output and returns output as character vector
Register using FNIRT, but doing Affine Registration as well

Description

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

Usage

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

Arguments

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

Value

character or logical depending on intern
fnirt_with_affine_apply

Applies FLIRT then FNIRT transformations

Description

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

Usage

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

Arguments

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

Value

character or logical depending on intern

See Also

fnirt_with_affine
fslabs.help 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 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**

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

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector.

**Examples**

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

---

**fslasin.help**  

---

**fslasin Help**

---

**Description**

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

**Usage**

```r
fslasin.help(...)  
```

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector.

**Examples**

```r
if (have.fsl()){
  fslasin.help()
}
```
**fslatan.help**  
*fslatan Help*

**Description**

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

**Usage**

```r
fslatan.help(...)  
```

**Arguments**

... passed to *fslmaths.help*.

**Value**

Prints help output and returns output as character vector.

**Examples**

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

**fslbet.help**  
*Help for FSL BET*

**Description**

This function calls bet's help.

**Usage**

```r
fslbet.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()){
  fslbet.help()
  fslbet.help("bet")
}

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

Usage
fslbin.help(...)  

Arguments
... passed to fslmaths.help

Value
Prints help output and returns output as character vector

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

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

Usage
fslbinv.help(...)  

Arguments
... passed to fslmaths.help
Value
Prints help output and returns output as character vector

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

fslchfiletype

FSL Change file type

Description
This function calls fslchfiletype

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

Arguments
  file (character) image to be manipulated
  filetype 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) print out command before running
  ... additional arguments passed to readnii.

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

This function calls fslchfiletype’s help

Usage

fslchfiletype.help()

Value

Prints help output and returns output as character vector

Examples

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

fslcmd

FSL Command Wrapper

Description

This function calls fsl command passed to func

Usage

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

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
fslcog

**Image Center of Gravity (FSL)**

**Description**

Find Center of Gravity of Image from FSL

**Usage**

```r
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**

```r
if (have.fsl()){
x = array(rnorm(1e6), dim = c(100, 100, 100))
img = nifti(x, dim = c(100, 100, 100),
datatype = convert.datatype(char2$FLOAT32, cal.min = min(x),
cal.max = max(x), pixdim = rep(1, 4))
fslcog(img)
}
```
\textbf{fslcos.help} \hspace{1cm} \textit{fslcos Help}

\textbf{Description}

This function calls \texttt{fslmaths}'s help, as \texttt{fslcos} is a wrapper for \texttt{fslmaths}.

\textbf{Usage}

\begin{verbatim}
   fslcos.help(...)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
   \item \texttt{...} passed to \texttt{fslmaths.help}
\end{itemize}

\textbf{Value}

Prints help output and returns output as character vector.

\textbf{Examples}

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

---

\textbf{fslcpgeom} \hspace{1cm} \textit{FSL Copy Geometry}

\textbf{Description}

This function calls \texttt{fslcpgeom}.

\textbf{Usage}

\begin{verbatim}
   fslcpgeom(file, file_with_header, retimg = TRUE, reorient = FALSE,
                  intern = FALSE, opts = "", verbose = TRUE, ...)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
   \item \texttt{file} (character) image to be manipulated
   \item \texttt{file_with_header} image with header to be copied over
   \item \texttt{retimg} (logical) return image of class nifti
   \item \texttt{reorient} (logical) If retimg, should file be reoriented when read in? Passed to \texttt{readnii}.
   \item \texttt{intern} (logical) to be passed to \texttt{system}.
\end{itemize}
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

Description
This function calls fslcpgeom’s help

Usage
fslcpgeom.help()

Value
Prints help output and returns output as character vector

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

fsldir

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

Usage
fsldir()

Value
Character path
**fsldiv.help**

**fsldiv Help**

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

**Usage**

fsldiv.help(...)

**Arguments**

... passed to fslmaths.help

**Value**

Prints help output and returns output as character vector

**Examples**

```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**

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

Describe Image Mean Entropy

Description

Estimates Mean Entropy of Image from FSL.

Usage

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

fslerode.help

fslerode Help

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
**fslexp.help**

**Examples**

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

---

**Description**

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

**Usage**

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

**Arguments**

... passed to fslmaths.help

**Value**

Prints help output and returns output as character vector

**Examples**

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

---

**fslfill.help**

**fslfill Help**

---

**Description**

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

**Usage**

```r
fslfill.help(...) 
```

**Arguments**

... passed to fslmaths.help
Value

Prints help output and returns output as character vector

Examples

```r
if (have.fsl()){
  fslfill2.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, reting = 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.
- **reting** (logical) return image of class nifti
- **reorient** (logical) If reting, should file be reoriented when read in? Passed to readnii.
- **intern** (logical) pass to system
- **verbose** (logical) print out command before running
- **...** additional arguments passed to readnii.

Value

character or logical depending on intern

Note

This function binarizes the image before running.
fslgetorient  

**FSL Orientation Wrappers**

**Description**

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

**Usage**

```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

if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard",
  "MN152_T1_2mm.nii.gz")
  fslhd(mnifile)
}  

fslhd.help

Description

This function calls fslhd’s help

Usage

fslhd.help()

Value

Prints help output and returns output as character vector

Examples

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

fslhd.parse

Description

This function takes in a FSL header and parses the components

Usage

fslhd.parse(hd)

Arguments

hd (character) header from fslhd
Value
data.frame of information from FSL header

Examples
if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard",
                    "MN152_T1_2mm.nii.gz")
  hd = fslhd(mnifile)
  fslhd.parse(hd)
}

---

fslhelp

Wrapper for getting fsl help

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

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

Arguments

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

Value
Prints help output and returns output as character vector

---

fslindex.help

fslindex Help

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

Usage
fslindex.help(...)
Arguments

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

Value

Prints help output and returns output as character vector

Examples

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

**fslmask Help**

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

### Usage
```
fslmask.help()
```

### Arguments
```
...
```
passed to `fslmaths.help`

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

### Examples
```
if (haveFsl()){
  fslmask.help()
}
```

---

fslmaths.help  

**FSL Maths Help**

### Description
This function calls fslmaths's help

### Usage
```
fslmaths.help()
```

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

### Examples
```
if (haveFsl()){
  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
- `...` 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
**fslmerge.help**

**Value**
Vector of unless ts option invoked, then matrix

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

---

**fslmerge.help**

**FSLMerge help**

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

**Usage**
fslmerge.help()

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

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

---

**fslmul.help**

**fslmul Help**

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

**Usage**
fslmul.help(...)

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

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

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

---

**fslnanm.help**

### Description

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

### Usage

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

#### Arguments

... passed to `fslmaths.help`

### Value

Prints help output and returns output as character vector

---

**fslnanm Help**

### Description

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

### Usage

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

#### Arguments

... passed to `fslmaths.help`
Value

Prints help output and returns output as character vector

Examples

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

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.
**fslorienter**  

*Wrapper for FSL Get Orientation*

**Description**

This function calls fslorient's help

**Usage**

```r
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**

```r
fslrand.help(...)  
```

**Arguments**

... passed to `fslmaths.help`

**Value**

Prints help output and returns output as character vector.

**Examples**

```r
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

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

Arguments

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

Value

numeric vector of length 2

Examples

```r
if (have.fsl()){
  mnifile = file.path(fsldir(), "data", "standard",
                     "MN152_T1_2mm.nii.gz")
  fslrange(mnifile)
}
```

---

fslrecip.help  

fslrecip Help

Description

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

Usage

fslrecip.help(...)

Arguments

... passed to fslmaths.help
Value

Prints help output and returns output as character vector

Examples

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

---

**fslrem Help**

Description

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

Usage

```r
fslrem.help(...)```

Arguments

... passed to `fslmaths.help`

Value

Prints help output and returns output as character vector

Examples

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

---

**fslreorient2std FSL Orient to MNI**

Description

This function calls fslreorient2std

Usage

```r
def 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.

Description

This function calls `fslreorient2std`'s help

Usage

`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, ...)
```

```r
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()
```
Description

This function calls fslroi

Usage

\[ fslroi(file, xmin = NULL, xsize = NULL, ymin = NULL, ysize = NULL, 
    zmin = NULL, zsize = NULL, tmin = NULL, tsize = NULL, 
    outfile = NULL, retimg = TRUE, reorient = FALSE, intern = FALSE, 
    verbose = TRUE, ...) \]

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**

\[
fslsd(img, \text{nonzero} = \text{FALSE}, \text{verbose} = \text{TRUE}, \text{ts} = \text{FALSE})
\]

**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**

\[
fslsin(file, \text{outfile} = \text{NULL}, \text{retimg} = \text{TRUE}, \text{reorient} = \text{FALSE}, \\
\qquad \text{intern} = \text{FALSE}, \text{opts} = "", \ldots)
\]

**Arguments**

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

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

---

fslsin.help  

**fslsin Help**

Description

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

Usage

```
fslsin.help(...)```

Arguments

... passed to `fslmaths.help`

Value

Prints help output and returns output as character vector

Examples

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

---

fslslicetimer  

**FSL Slice Timing Correction**

Description

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

Usage

```
fslslicetimer(file, outfile = NULL, retimg = TRUE, reorient = FALSE, intern = FALSE, tr = 3L, 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)
- **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`
- **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 `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

---

**fslsmooth.help** *fslsmooth Help*

Description

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

Usage

`fslsmooth.help(...)`

Arguments

... passed to `fslmaths.help`

Value

Prints help output and returns output as character vector
Examples

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

def fslsmooth_in_mask(file, sigma = 10, 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

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("$\text{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**

```r
fslsplit(infile, direction = c("t", "x", "y", "z"), output_basename = NULL, 
retimg = TRUE, reorient = FALSE, verbose = TRUE)
```

```r
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  

**Description**

This function calls fslsplit's help

**Usage**

fslsplit.help()

**Value**

Prints help output and returns output as character vector

**Examples**

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

---

fslsqr.help  

**Description**

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

**Usage**

fslsqr.help(...)

**Arguments**

... passed to fslmaths.help

**Value**

Prints help output and returns output as character vector

**Examples**

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

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

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

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

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

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

---

**fslstats**

**FSL Stats**

**Description**
This function calls fslstats.

**Usage**
```r
fslstats(file, opts = "", verbose = TRUE, ts = FALSE, ...) 
```

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

Result of fslstats command

Examples

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

---

**fslstats.help**

### 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**

### Description

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

### Usage

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

... passed to \texttt{fsl\textup{\textunderscore}maths\textunderscore}help

Value

Prints help output and returns output as character vector

Examples

\begin{verbatim}
if (have.fsl()){
    fslsub2.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()
}
```
**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()
}
```

---

**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()
}
```
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",
                          "MNI152_T1_2mm.nii.gz")
    fslval(mnifile, keyword = "dim1")
}

Description

This function calls fslval's help

Usage

fslval.help()

Value

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

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

---

**fslview**  
*Open image in FSLView*

---

**Description**

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

**Usage**

```r
def 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 `checkimg`).

**Value**

Character or logical depending on `intern`.

---

**fslview.help**  
*FSLView help*

---

**Description**

This function calls `fslview`'s help.

**Usage**

```r
fslview.help()
```

**Value**

Prints help output and returns output as character vector.

**Examples**

```r
if (have.fsl()){
  fslview.help()
}
```
### fslvol

**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`.

---

**fsl_abs**  
*Absolute Value Image using FSL*

**Description**

This function calls `fslmaths Mabs`. The R functions wraps `fslmaths`.

**Usage**

```r
fsl_abs(..., outfile = tempfile(fileext = "Nnii.gz"), retimg = FALSE)
fslabs(file, outfile = NULL, retimg = TRUE, reorient = FALSE, intern = FALSE, opts = "\"", ...)
```

**Arguments**

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

**Value**

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

**Note**

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

Arc Cosine Transform Image using FSL

Description

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

Usage

fsl_acos(..., outfile = tempfile(fileext = "nii.gz"), retrim = FALSE)
fslacos(file, outfile = NULL, retrim = TRUE, reorient = FALSE, intern = FALSE, opts = "", ...)

Arguments

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
retrim (logical) return image of class nifti
file (character) input image to arc cosine transform
reorient (logical) If retrim, 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 retrim 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 function wraps fslmaths.
Usage

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

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

Arguments

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

Value

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

Note

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

---

fsl_anat

FSL Anatomical Processing Script

Description

This function calls fsl_anat from FSL

Usage

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

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

Value

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

Description

This function calls fsl_anat's help

Usage

```r
fsl_anat.help()
```

Value

Prints help output and returns output as character vector

Examples

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

This function applies a coefficient map from \texttt{fnirt} to other images.

Usage

\begin{verbatim}
fsl_applywarp(infile, reffile, warpfile, outfile = NULL, retimg = TRUE, 
   reorient = FALSE, intern = FALSE, opts = "", verbose = TRUE, ...)
\end{verbatim}

Arguments

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

Value

character or logical depending on \texttt{intern}

Description

This function calls \texttt{applywarp}'s help.

Usage

\begin{verbatim}
fsl_applywarp.help()
\end{verbatim}

Value

Prints help output and returns output as character vector
Description

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

Usage

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

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

Arguments

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

Value

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

Note

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

Description

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

fsl_atan(..., outfile = tempfile(fileext = "nii.gz"), 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

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
def fsl_avscale(file, volume = NULL, parsed = TRUE, verbose = TRUE):
    avscale(...)```

**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*

---

fsl_bet  

*Use FSL’s Brain Extraction Tool (BET)*

**Description**

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

**Usage**

```r
def 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 \texttt{readnii}.
\textbf{outfile} (character) output filename
\textbf{retimg} (logical) return image of class nifti
\textbf{infile} (character) input filename
\textbf{reorient} (logical) If retimg, should file be reoriented when read in? Passed to \texttt{readnii}.
\textbf{intern} (logical) pass to \texttt{system}
\textbf{opts} (character) additional options to bet
\textbf{betcmd} (character) Use bet or bet2 function
\textbf{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

---

\textbf{fsl\_biascorrect} \hspace{1cm} \textit{FSL Bias Correct}

Description

This function wraps a call to \texttt{fast} that performs bias correction

Usage

\texttt{fsl\_biascorrect(file, outfile = NULL, retimg = TRUE, reorient = FALSE,}
\hspace{1cm} \texttt{intern = FALSE, opts = "", verbose = TRUE, remove.seg = TRUE, ...)}

Arguments

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

Binarize Image using FSL

Description

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

Usage

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

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

Arguments

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

Value

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

Note

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

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

**fsl_binv**

Binarized Inverse Image using FSL

**Description**

This function calls fslmaths `~binv`. The R functions wraps fslmaths.

**Usage**

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

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

**Arguments**

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

**Value**

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

**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)
```

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

... additional arguments passed to readnii.
outfile (character) resultant image name (optional)
reoting (logical) return image of class nifti
file (character) input image to cosine transform
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

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

---

fsl_data_dir

Get FSL's Data Directory

Description

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

Usage

fsl_data_dir()

Value

Character path
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 `readnii`
- `outfile`: (character) resultant dilated image name
- `retimg`: (logical) return image of class nifti
- `file`: (character) image to be dilated
- `reorient`: (logical) If retimg, should file be reoriented when read in? Passed to `readnii`
fsl_div

Divide Images using FSL

Description

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

Usage

fsl_div(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
fsldiv(file, file2, outfile = NULL, retimg = TRUE, reorient = FALSE, intern = FALSE, opts = "", ...)
Arguments

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

Value

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

Note

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

---

`fsl_edge`  
*Edge Strength Image using FSL*

Description

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

Usage

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

```r
fsl_edge(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 estimate edge strength
- `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_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"), retimg = FALSE)
fsl_erode(file, outfile = NULL, retimg = TRUE, reorient = FALSE, intern = FALSE, kopts = "", opts = "", verbose = TRUE, ...)
```

Arguments

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

Value

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

Note

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

```r
if (have.fsl()){
  system.time(
    dims = c(50, 50, 20)
    x = array(rnorm(prod(dims)), dim = dims)
    img = nifti(x, dim= dims, 
    datatype = convert.datatype($FLOAT32, cal.min = min(x),
    cal.max = max(x), pixdim = rep(1, 4))
    mask = img > .5
    eroded = fslerode(mask, kopts = "-kernel boxv 5", retimg=TRUE)
  }
}
```

---

**fsl_exp**

*Exponentiate Image using FSL*

### Description

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

### Usage

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

### Arguments

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

### Value

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

### Note

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

*Fill image holes*

**Description**

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

**Usage**

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

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

**Arguments**

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

**Value**

character or logical depending on `intern`.

**Note**

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

**Examples**

```r
if (have.fsl()){
  system.time(
    dims = c(50, 50, 20)
    x = array(rnorm(prod(dims)), dim = dims)
    img = nifti(x, dim= dims,
    datatype = convert.datatype<double>,$FLOAT32, cal.min = min(x),
    cal.max = max(x), pixdim = rep(1, 4))
    mask = img > .5
    eroded = fslerode(mask, kopts = "-kernel boxv 5", retimg=TRUE)
  }
```
This function calls `fslmaths -index`. The R function wraps `fslmaths`.

**Usage**

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

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

**Arguments**

- `...` additional arguments passed to `readnii`.
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `file` (character) input image to have non-zero entries replaced with index
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- `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 \texttt{fslmaths Mlog}. The R functions wraps \texttt{fslmaths}

**Usage**

\texttt{fsl_log(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)}

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

**Arguments**

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

**Value**

If \texttt{retimg} then object of class nifti. Otherwise, Result from system command, depends if \texttt{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 \texttt{fslmaths Mmas} to mask an image from an image mask and either saves the image or returns an object of class nifti
Usage

\[
\text{fsl\_mask}(\ldots, \text{outfile} = \text{tempfile(fileext} = "\text{nii.gz}")\text{), retimg} = \text{FALSE})
\]

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

Arguments

\[
\ldots \quad \text{additional arguments passed to readnii.}
\]

\[
\text{outfile} \quad \text{(character) resultant masked image name}
\]

\[
\text{retimg} \quad \text{(logical) return image of class nifti}
\]

\[
\text{file} \quad \text{(character) image to be masked}
\]

\[
\text{mask} \quad \text{(character) mask given for image}
\]

\[
\text{reorient} \quad \text{(logical) If retimg, should file be reoriented when read in? Passed to readnii.}
\]

\[
\text{intern} \quad \text{(logical) to be passed to system}
\]

\[
\text{opts} \quad \text{(character) additional options to be passed to fslmask}
\]

\[
\text{verbose} \quad \text{(logical) print out command before running}
\]

Value

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

Note

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

\[
\text{fsl\_maths}
\]

\[
FSL \text{ Maths}
\]

Description

This function calls fslmaths

Usage

\[
\text{fsl\_maths}(\ldots, \text{outfile} = \text{tempfile(fileext} = "\text{nii.gz}")\text{), retimg} = \text{FALSE})
\]

\[
\text{fsl\_maths(file, outfile} = \text{NULL, retimg} = \text{TRUE, reorient} = \text{FALSE, intern} = \text{FALSE, opts} = "", \text{verbose} = \text{TRUE, \ldots})
\]
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` operations to be passed to `fslmaths`
- `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_merge

**Merge images using FSL**

**Description**

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

**Usage**

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

fslmerge(infiles, direction = c("x", "y", "z", "t", "a"), outfile = NULL, retimg = TRUE, reorient = FALSE, intern = FALSE, verbose = TRUE, ...)```

**Arguments**

- `...`: additional arguments passed to `readnii`
- `outfile` (character): output filename
- `retimg` (logical): return image of class nifti
- `infiles` (character): input filenames
- `direction` (character): direction to merge over, x, y, z, t (time), a (auto)
- `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
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

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_nan**

Replace NaNs in Image using FSL

**Description**

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

**Usage**

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

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

Arguments

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

Value

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

Note

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

---

fsl_rand

Add Random Uniform Noise Image using FSL

Description

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

Usage

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

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

... additional arguments passed to \texttt{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 \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 intern is TRUE or FALSE.

Note

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

\begin{center}
\begin{longtable}{ll}
\hline
\texttt{fsl\_randn} & \textit{Add Random Standard Guassian Noise Image using FSL} \\
\hline
\end{longtable}
\end{center}

Description

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

Usage

\begin{verbatim}
fsl\_randn(..., outfile = tempfile(fileext = ".nii.gz"), reorient = FALSE)
fsl\_randn(file, outfile = NULL, reorient = TRUE, reorient = FALSE,
        intern = FALSE, opts = "$" , ...)
\end{verbatim}

Arguments

... additional arguments passed to \texttt{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 \texttt{readnii}.
intern (logical) to be passed to \texttt{system}
opts (character) operations to be passed to \texttt{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.
Modulus Remainder of 2 Images using FSL

Description

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

Usage

```r
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**

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

fslsmooth(file, sigma = 10, mask = NULL, smooth_mask = TRUE,
          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 given, will be the stub of the filename then `_sigma`.
- `retimg`: (logical) return image of class nifti.
- `file`: (character) 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.
- `reorient`: (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- `intern`: (logical) to be passed to `system`.
- `verbose`: (logical) print out command before running.

**Value**

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

**Note**

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

```r
if (have.fsl()){
  system.time(
    dims = c(50, 50, 20)
    x = array(rnorm(prod(dims)), dim = dims)
    img = nifti(x, dim= dims,
    datatype = convert.datatype('&$FLOAT32', cal.min = min(x),
    cal.max = max(x), pixdim = rep(1, 4))
    s.img = fslsmooth(img, retimg=TRUE)
  )
}
```

---

**fsl_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)
fsl.sqrt(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("fslNpath") and pastes on “data/standard”

Usage
fsl_std_dir()
**fsl_sub**

*Subtract Images using FSL*

**Value**

Character path

---

**Description**

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

**Usage**

```r
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**

```r
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**

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

```r
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 function wraps `fslmaths`.

**Usage**

```r
fsl_tan(..., outfile = tempfile(fileext = "Nnii.gz"), retimg = FALSE)
```

```r
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

\[
\text{fsl\_thresh(..., outfile = tempfile(fileext = ".nii.gz"), retimg = FALSE)
}\]

\[
\text{fslthresh(file, outfile = NULL, thresh = 0, uthresh = NULL, retimg = TRUE, reorient = FALSE, intern = FALSE, opts = ",", verbose = TRUE, ...)
}\]

Arguments

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

Value

character or logical depending on \texttt{intern}

Note

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

Examples

\[
\text{if (have.fsl())}{
\text{system.time({}
\text{x = array(rnorm(1e6), dim = c(100, 100, 100))}
\text{img = nifti(x, dim= c(100, 100, 100),
\text{datatype = convert.datatype()$FLOAT32, cal.min = min(x),
\text{cal.max = max(x), pixdim = rep(1, 4))
\text{thresh = fslthresh(img, thresh=0, uthresh = 2, retimg=TRUE)}
\text{)}}
\text{})}
\}
fsl_version  
\textit{Find FSL Version}

\textbf{Description}

Finds the FSL version from FSLDIR/etc/fslversion

\textbf{Usage}

\begin{verbatim}
fsl_version()
\end{verbatim}

\textbf{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.

\textbf{Note}

This will use fsldir() to get the directory

\textbf{Examples}

\begin{verbatim}
if (have_fsl()) {
  fslversion()
  fsl_version()
}
\end{verbatim}

---

get.fsl  
\textit{Create command declaring FSLDIR}

\textbf{Description}

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

\textbf{Usage}

\begin{verbatim}
get.fsl(add_bin = TRUE)
\end{verbatim}

\textbf{Arguments}

\begin{verbatim}
add_bin  Should bin be added to the fsl path? All executables are assumed to be in FSLDIR/bin/. If not, and add_bin = FALSE, they will be assumed to be in FSLDIR/.
\end{verbatim}
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

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

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

getForms(file, ...)

Arguments

file  
(character) filename of image to pass to header

...

options passed to checkimg

Value

list with elements of sform and qform and their respective codes

Examples

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

have.fsl(...) 

have_fsl(...) 

Arguments

...

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

Examples
have.fsl()

Description
intent_code method for character types

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

Arguments
object is a filename to pass to fslval

Description
intent_name method for character types

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

Arguments
object is a filename to pass to fslval
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  

**Description**

This function calls mcflirt's help

**Usage**

mcflirt.help()

**Value**

Prints help output and returns output as character vector

**Examples**

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

melodic  

**Description**

This function calls melodic

**Usage**

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

**Arguments**

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

**Value**

character or logical depending on intern
**Description**

This function calls melodic’s help

**Usage**

melodic.help()

**Value**

Prints help output and returns output as character vector

**Examples**

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

---

**mid_sagittal_align**  *Mid-Sagittal Plane Alignment*

Description

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

**Usage**

```r
mid_sagittal_align(fileL outfile = NULLL retimg = TRUEL 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`
\texttt{mni\_fname} \hspace{1cm} \textit{Construct MNI Filename}

\textbf{Description}

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

\textbf{Usage}

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

\textbf{Arguments}

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

\textbf{Value}

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

\texttt{mni\_img} \hspace{1cm} \textit{Read MNI Filename}

\textbf{Description}

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

\textbf{Usage}

\texttt{mni\_img(\ldots)}

\textbf{Arguments}

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

\textbf{Value}

Object of class \texttt{nifti}
parse_avscale

Parse output from 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**

```r
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, target3 = 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
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.waycond</td>
<td>Waypoint condition. Either 'AND' (default) or 'OR'</td>
</tr>
<tr>
<td>.wayorder</td>
<td>Reject streamlines that do not hit waypoints in given order. Only valid if way-cond=AND</td>
</tr>
<tr>
<td>.onewaycondition</td>
<td>Apply waypoint conditions to each half tract separately</td>
</tr>
<tr>
<td>.avoid</td>
<td>Reject pathways passing through locations given by this mask</td>
</tr>
<tr>
<td>.stop</td>
<td>Stop tracking at locations given by this mask file</td>
</tr>
<tr>
<td>.omatrix1</td>
<td>Output matrix1 - SeedToSeed Connectivity</td>
</tr>
<tr>
<td>.distthresh1</td>
<td>Discards samples (in matrix1) shorter than this threshold (in mm - default=0)</td>
</tr>
<tr>
<td>.omatrix2</td>
<td>Output matrix2 - SeedToLowResMask</td>
</tr>
<tr>
<td>.target2</td>
<td>Low resolution binary brain mask for storing connectivity distribution in matrix2</td>
</tr>
<tr>
<td>.mode</td>
<td>Apply waypoint conditions to each half tract separately</td>
</tr>
<tr>
<td>.omatrix3</td>
<td>Output matrix3 (NxN connectivity matrix)</td>
</tr>
<tr>
<td>.target3</td>
<td>Mask used for NxN connectivity matrix (or Nxn if lrtarget3 is set)</td>
</tr>
<tr>
<td>.lrtarget3</td>
<td>Column-space mask used for Nx connectivity matrix</td>
</tr>
<tr>
<td>.distthresh3</td>
<td>Discards samples (in matrix3) shorter than this threshold (in mm - default=0)</td>
</tr>
<tr>
<td>.xfm</td>
<td>Transform taking seed space to DTI space (either FLIRT matrix or FNIRT warp-field) - default is identity</td>
</tr>
<tr>
<td>.invxfm</td>
<td>Transform taking DTI space to seed space (compulsory when using a warpfield for seeds_to_dti)</td>
</tr>
<tr>
<td>.seedref</td>
<td>Reference vol to define seed space in simple mode - diffusion space assumed if absent</td>
</tr>
<tr>
<td>.meshspace</td>
<td>Mesh reference space - either 'caret' (default) or 'freesurfer' or 'first' or 'vox'</td>
</tr>
<tr>
<td>.nsamples</td>
<td>Number of samples - default=5000</td>
</tr>
<tr>
<td>.nsteps</td>
<td>Number of steps per sample - default=2000</td>
</tr>
<tr>
<td>.steplength</td>
<td>Steplength in mm - default=0.5</td>
</tr>
<tr>
<td>.distthresh</td>
<td>Discards samples shorter than this threshold (in mm - default=0)</td>
</tr>
<tr>
<td>.cthr</td>
<td>Curvature threshold - default=0.2</td>
</tr>
<tr>
<td>.fibthresh</td>
<td>Volume fraction before subsidiary fibre orientations are considered - default=0.01</td>
</tr>
<tr>
<td>.loopcheck</td>
<td>Perform loopchecks on paths - slower, but allows lower curvature threshold</td>
</tr>
<tr>
<td>.usef</td>
<td>Use anisotropy to constrain tracking</td>
</tr>
<tr>
<td>.modeuler</td>
<td>Use modified euler streamlining</td>
</tr>
<tr>
<td>.sampvox</td>
<td>Sample random points within x mm sphere voxels (e.g. --sampvox=5).</td>
</tr>
<tr>
<td>.randfib</td>
<td>Default 0. Set to 1 to randomly sample initial fibres (with f &gt; 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&lt;fibthresh)</td>
</tr>
<tr>
<td>.fibst</td>
<td>Force a starting fibre for tracking - default=1, i.e. first fibre orientation. Only works if randfib==0</td>
</tr>
<tr>
<td>.rseed</td>
<td>Random seed</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments</td>
</tr>
</tbody>
</table>
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)
```

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

Arguments

- `object` is a `nifti` object

qform_code-method

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

**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_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)
```

```r
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, 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

Result of system

---

run_first_all.help | Run FIRST All Help

Description

This function calls run_first_all’s help

Usage

run_first_all.help()

Value

Prints help output and returns output as character vector

Examples

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'
sizeof_hdr(object)
```

**Arguments**

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

---

### `slice_code` methods

**Description**

`slice_code` method for character types

**Usage**

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

**Arguments**

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

*Extract Image slice_duration attribute*

Description

slice_duration method for character types

Usage

```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**

```r
## 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 Univalence Segment Assimilating Nucleus (SUSAN) noise reduction technique from FSL

**Usage**

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

**Arguments**

- `file` (character) image to be manipulated
- `outfile` (character) resultant image name (optional)
- `retimg` (logical) return image of class nifti
- `reorient` (logical) If retimg, should file be reoriented when read in? Passed to `readnii`.
- `intern` (logical) to be passed to `system`
- `bthresh` brightness threshold and should be greater than noise level and less than contrast of edges to be preserved.
- `sigma` spatial size (sigma i.e. half-width) of smoothing in mm.
- `dimg` dimensionality (2 or 3) depending on whether smoothing is to be within-plane (2) or fully 3D (3).
use_median determines whether to use a local median filter in the cases where single-point noise is detected (0 or 1).

n_usans determines whether the smoothing area (USAN) is to be found from secondary images (0 1 or 2).

extra.scans List of extra scans for USAN. List of n_usans elements, where each element has 2 named objects bthresh and filename

opts (character) operations to be passed to susan, not currently used.

verbose (logical) print out command before running

... additional arguments passed to fslcmd.

Value

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

References


Description

This function calls susan’s help

Usage

susan.help()

Value

Prints help output and returns output as character vector

Examples

if (have.fsl()){
  susan.help()
}
**toffset-methods**

*Extract Image toffset attribute*

### Description

Gets toffset from a character

### Usage

```r
## S4 method for signature 'character'
toffset(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(imain, datain, out = NULL, fout = NULL, iout = NULL,
      logout = NULL, warpres = 10, subsamp = 1, fwhm = 8, config = NULL,
      miter = 5, lambda = NULL, ssqlambda = 1, regmod = c("bending_energy",
      "membrane_energy"), estmov = 1, minmet = c(0, 1), splineorder = c(3, 2),
      numprec = c("double", "float"), interp = c("spline", "linear"),
      scale = c(0, 1), regrid = c(0, 1), verbose = TRUE)
```

```r
fsl_topup(...)```

### Arguments

- **imain**
  - 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
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subsamp</td>
<td>sub-sampling scheme, default 1</td>
</tr>
<tr>
<td>fwhm</td>
<td>FWHM (in mm) of gaussian smoothing kernel, default 8</td>
</tr>
<tr>
<td>config</td>
<td>Name of config file specifying command line arguments</td>
</tr>
<tr>
<td>miter</td>
<td>Max # of non-linear iterations, default 5</td>
</tr>
<tr>
<td>lambda</td>
<td>Weight of regularisation, default depending on ssqlambda and regmod switches. See user documentation.</td>
</tr>
<tr>
<td>ssqlambda</td>
<td>If set (=1), lambda is weighted by current ssq, default 1</td>
</tr>
<tr>
<td>regmod</td>
<td>Model for regularisation of warp-field [membrane_energy bending_energy], default bending_energy</td>
</tr>
<tr>
<td>estmov</td>
<td>Estimate movements if set, default 1 (true)</td>
</tr>
<tr>
<td>minmet</td>
<td>Minimisation method 0=Levenberg-Marquardt, 1=Scaled Conjugate Gradient, default 0 (LM)</td>
</tr>
<tr>
<td>splineorder</td>
<td>Order of spline, 2-&gt;Quadratic spline, 3-&gt;Cubic spline. Default=3</td>
</tr>
<tr>
<td>numprec</td>
<td>Precision for representing Hessian, double or float. Default double</td>
</tr>
<tr>
<td>interp</td>
<td>Image interpolation model, linear or spline. Default spline</td>
</tr>
<tr>
<td>scale</td>
<td>If set (=1), the images are individually scaled to a common mean, default 0 (false)</td>
</tr>
<tr>
<td>regrid</td>
<td>If set (=1), the calculations are done in a different grid, default 1 (true)</td>
</tr>
<tr>
<td>verbose</td>
<td>Print diagnostic information while running</td>
</tr>
<tr>
<td>...</td>
<td>arguments passed to topup if using fsl_topup</td>
</tr>
</tbody>
</table>

### Description

`vox_offset` method for character types

### Usage

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

### Arguments

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

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

**Description**

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

**Usage**

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

**Arguments**

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

Use ard on f0

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), 68

bedpostx (xfibres), 121
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, 31, 42, 53, 58, 59, 64, 100, 104
checkout, 7, 8

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

dddy_correct, 11

fast, 12
fast.help, 12
flirt, 13, 14, 105
flirt.help, 14

fsl_abs, 61
fsl_acos, 62
fsl_add, 62
fsl_anat, 63
fsl_anat.help, 64
fsl_applywarp, 65
fsl_applywarp.help, 65
fsl_asin, 66
fsl_atan, 66
fsl_atlas_dir, 67
fsl_avscale, 68, 107
fsl_bet, 68
fsl_biascorrect, 69
fsl_bin, 70
fsl_bin_tab, 72, 74
fsl_binw, 71
fsl_cos, 72
fsl_data_dir, 73
fsl_dice, 74
fsl_dilate, 74
fsl_dir (fsldir), 26
fsl_div, 75
fsl_edge, 76
fsl_erode, 77
fsl_exp, 78
fsl_fill, 79
fsl_index, 80
fsl_log, 81
fsl_mask, 81
fsl_maths, 82
fsl_merge, 83
fsl_mul, 84
fsl_nan, 85

fnirt, 15, 17, 65
fnirt.help, 15
fnirt_with_affine, 16
fnirt_with_affine_apply, 17

123
fsl_nanm, 85
fsl_rand, 86
fsl_randn, 87
fsl_recip, 88
fsl_rem, 89
fsl_robustfov (fslrobustfov), 45
fsl_slicetimer (fslslicetimer), 48
fsl_smooth, 90
fsl_smooth_in_mask (fslsmooth_in_mask), 50
fsl_split (fslsplit), 51
fsl_sqr, 91
fsl_sqrt, 92
fsl_std_dir, 92
fsl_sub, 93
fsl_sub2, 94
fsl_swapdim, 95
fsl_tan, 96
fsl_thresh, 96
fsl_topup (topup), 119
fsl_version, 98
fslabs (fsl_abs), 61
fslabs.help, 18
fslacos (fsl_acos), 62
fslacos.help, 18
fsladd (fsl_add), 62
fsladd.help, 19
fslasin (fsl_asin), 66
fslasin.help, 19
fslatan (fsl_atan), 66
fslatan.help, 20
fslbet, \[11, 121\]
fslbet (fsl_bet), 68
fslbet.help, 20
fslbin (fsl_bin), 70
fslbin.help, 21
fslbinv (fsl_inv), 71
fslbinv.help, 21
fslchfiletype, 22
fslchfiletype.help, 23
fslcmd, \[11, 23, 118\]
fslcog, 24
fslcos (fsl_cos), 72
fslcos.help, 25
flspgeom, 25
flspgeom.help, 26
fsldivide (fsl_dilate), 74
fsldir, 26
fsldiv (fsl_div), 75
fsldiv.help, 27
fslledge (fsl_edge), 76
fslledge.help, 27
fslentropy, 28
fslentropy.help (fslstats.help), 54
fslerode (fsl_erode), 77
fslerode.help, 28
fslexp (fsl_exp), 78
fslexp.help, 29
fslfill, 30
fslfill (fsl_fill), 79
fslfill.help, 29
fslfill2, 30
fslgetorient, 31
fslgetqform (fslgetorient), 31
fslgetqformcode (fslgetorient), 31
fslgetstats (fslgetorient), 31
fslhd, \[31, 32\]
fslhd.help, 32
fslhd.parse, 32
fslhelp, 33
fslindex (fsl_index), 80
fslindex.help, 33
fsllog (fsl_log), 81
fsllog.help, 34
fslmask (fsl_mask), 81
fslmask.help, 35
fslmaths (fsl_maths), 82
fslmaths.help, \[18–21, 25, 27–29, 34, 35, 37, 38, 41–43, 48, 49, 52, 53, 55, 57\]
fslnan, 36
fslnan.help (fslstats.help), 54
fslmerge (fsl_merge), 83
fslmerge.help, 37
fslmax, 36
fslmean, 36
fslmean.help (fslstats.help), 54
fslmin (fsl_min), 36
fslmul (fsl_mul), 84
fslmul.help, 37
fslnan (fsl_nan), 85
fslnan.help, 38
fslnanm (fsl_nanm), 85
fslnanm.help, 38
fslorient, \[31, 39, 40\]
fslorient.help, 40
fslorienter, 40
fslrand (fsl_rand), 86
qform_code, character-method
  (qform_code-methods), 110
qform_code-methods, 110
readrpi, 111
reverse_rpi_orient, 111
reverse_rpi_orient_file
  (reverse_rpi_orient), 111
rpi_orient, 111, 112
rpi_orient_file, 111
rpi_orient_file(rpi_orient), 112
run_first_all, 112
run_first_all.help, 113
scl_inter, character-method
  (scl_inter-methods), 114
scl_inter-methods, 114
scl_slope, character-method
  (scl_slope-methods), 114
scl_slope-methods, 114
sform, character
  (qform, character-method), 110
sform, character-method
  (qform, character-method), 110
sform_code, character-method
  (sform_code-methods), 114
sform_code-methods, 114
sizeof_hdr, character-method
  (sizeof_hdr-methods), 115
sizeof_hdr-methods, 115
slice_code, character-method
  (slice_code-methods), 115
slice_code-methods, 115
slice_duration, character-method
  (slice_duration-methods), 116
slice_duration-methods, 116
slice_end, character-method
  (slice_end-methods), 116
slice_end-methods, 116
slice_start, character-method
  (slice_start-methods), 117
slice_start-methods, 117
susan, 117
susan.help, 118
system, 12–17, 22, 23, 25, 30, 39, 44–47, 49, 59, 61–67, 69–71, 73, 75–97, 103,

toffset, character-method
  (toffset-methods), 119
toffset-methods, 119
topup, 119
vox_offset, character-method
  (vox_offset-methods), 120
vox_offset-methods, 120
voxdim, 60
xfibres, 121