Package ‘arf3DS4’

February 19, 2015

Type Package
Title Activated Region Fitting, fMRI data analysis (3D).
Version 2.5-10
Date 2014-02-22
Author Wouter D. Weeda
Maintainer Wouter D. Weeda <w.d.weeda@gmail.com>
Depends R (>= 2.15.0), methods, graphics, tcltk, corpcor
Description Activated Region Fitting (ARF) is an analysis method for fMRI data.
URL http://arf.r-forge.r-project.org/
License GPL-2
LazyLoad yes
NeedsCompilation yes
Repository CRAN
Date/Publication 2014-02-21 11:29:26

R topics documented:

arf3DS4-package .................................................. 3
arf-example-data ................................................. 4
arfcorrelation-class .......................................... 5
arfversion ........................................................... 6
BIC ................................................................. 6
checkSolution ..................................................... 7
clearWarnings ................................................... 8
createAllAverages ............................................... 8
createAverages ................................................... 9
createReg .......................................................... 10
data-class ......................................................... 11
determineStartRect ........................................... 12
differenceCor ..................................................... 13
euclidDist ......................................................... 14
R topics documented:

- experiment-class .................................................. 14
- extr-repl-fmridata .................................................. 16
- fitConnectivity ..................................................... 17
- fitModel .............................................................. 18
- fitRange ............................................................... 19
- fmri.data-class ....................................................... 20
- fmri2array ............................................................. 22
- functional-class ...................................................... 23
- getAtlasLabels ....................................................... 24
- getExp ................................................................. 25
- getModelAtlas ......................................................... 25
- loadData ............................................................... 26
- loadExp ............................................................... 27
- loadFunc .............................................................. 28
- loadModel ............................................................ 28
- loadOptions .......................................................... 29
- loadRda ............................................................... 30
- loadReg .............................................................. 30
- loadStart ............................................................. 31
- makeDerivs ............................................................ 32
- makeExpDirs .......................................................... 32
- makeLowResStruct .................................................. 33
- makeROImask ......................................................... 34
- makeSingleTrialEvents ............................................. 35
- mcpCorrect .......................................................... 36
- minBIC ............................................................... 37
- mnames-class .......................................................... 37
- model-class .......................................................... 38
- newModel ............................................................. 40
- nifti.fileinfo-class ................................................ 41
- nifti.header-class .................................................... 42
- options-class ......................................................... 45
- processModel ........................................................ 46
- processSeed .......................................................... 47
- pruneModel .......................................................... 48
- readData .............................................................. 49
- readDerivs ............................................................ 50
- registration-class ................................................... 50
- RMSEA ............................................................... 52
- roiConnectivity ...................................................... 52
- saveFunc ............................................................. 53
- saveModel ............................................................ 54
- saveModelBin .......................................................... 54
- saveReg ............................................................... 55
- sequence-class ....................................................... 56
- setFuncTimings ....................................................... 57
- setIsoContour ........................................................ 58
- setRegFiles ........................................................... 58
Activated Region Fitting (ARF) is an analysis method for fMRI data. The ARF method uses Gaussian shaped functions to model active brain regions. An entire fMRI volume can be described by multiple parameterized Gaussian regions of activation. This parameterization allows for hypotheses on the location of an active region, the spatial extent of an active region, and the amplitude of an active region. ARF can be invoked after standard GLM analysis, and uses the beta-values or t-values from this analysis. In addition ARF can be used to estimate functional connectivity, for this the raw time-series (on which the GLM was performed) are also needed. The arf3DS4 package can read and write fMRI data in Nifti format (fmi.data). It is build to be compatible with fMRI analysis packages (e.g. FSL or SPM). The package uses a predefined directory- and file-structure to store its data. All objects (S4-class) and fMRI data files are stored in this directory- and file-structure.

Details

Package: arf3DS4
Type: Package
Version: 2.5-4
Date: 2011-10-28
Depends: R (>= 2.12.0), methods, graphics, tcltk
Repository/R-Forge/Revision: 203
Publication: 2011-10-28 12:38:50
Packaged: 2011-10-28 12:38:50; rforge
Built: R 2.12.0; universal-apple-darwin9.8.0; 2011-10-28 12:38:50; unix
LazyLoad: yes

Quick start

The main directory- and file-structure (termed an experiment) is handled by an object of class experiment. To make an empty experiment structure call makeExpDirs. This will create a directory-
and file-structure to hold the fMRI data (which will have to be copied manually to the appropriate directories, see data for an overview of the data-directories). Once this is done an experiment is loaded by calling loadExp. To fit the ARF models (class model) to the data, first create a model using newModel, adjust the options of the model via the options-object using loadOptions and saveOptions, and finally fit the model using fitModel. To perform hypothesis tests call varcov and wald to calculate the parameter (co)variance matrix and Wald statistics respectively. In addition to the hypothesis test connectivity between brain regions can be estimated (call fitConnectivity to perform this analysis). This function requires that the raw time-series of the fMRI data are available and that single-trial estimates (call makeSingleTrialEvents) are calculated.

Example data

The package comes with an example dataset (load it using data("arf-example-data")). By then calling makeExample, a directory-structure containing the example data is then created and loaded. Call getExp to view the data within the experiment.

Author(s)

Wouter D. Weeda <w.d.weeda@gmail.com>

References


arf-example-data  Activated Region Fitting Example Dataset

Description

Simulated dataset for the ARF3DS4 package. Run makeExample(path) to create an ARF directory structure in 'path'. The experiment is automatically loaded.

Usage

makeExample(path)

Arguments

path  Optional path to make the example experiment in. By default uses the path of the package installation.
arfcorrelation-class

Format
An ARF experiment for one subject ('wouter'), and one condition ('A').

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

Description
Object of class "arfcorrelation" hold information on correlations between regions of the spatial model. The functions fitConnectivity and roiConnectivity have as output an object of class "arfcorrelation".

Objects from the Class
Objects can be created by calls of the form new("arfcorrelation", ...).

Slots
- timebyreg: Trial-by-trial amplitude estimates per region (time x nreg)
- corr: Correlation matrix.
- corr.pval: P-values of the correlations
- pacorr: Partial correlation matrix (corrected for all other regions).
- pacorr.pval: P-values of partial correlations
- num.corr: Number of correlations tested.

Methods
Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

access .classname.slotname(object)
replace .classname.slotname(object)<-

In this case classname is "arfcorrelation" and slotname is for example "correlation". So to access the "correlation" slot of an r-object "arfcor" (of class "arfcorrelation") type .arfcorrelation.correlation(arfcor).

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

Examples
showClass("arfcorrelation")
arfversion  

Check the version of ARF objects.

Description

Print version information of an ARF object.

Usage

arfversion(object)

Arguments

object     

An arf object.

Value

None. The function prints the current version of the object.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

version

BIC  

Bayesian Information Criterion

Description

BIC calculates the Bayesian Information Criterion (model fit) for a valid ARF model. BIC requires a valid (converged) ARF model (`model`).

Usage

BIC(arfmodel, options = loadOptions(arfmodel))

Arguments

arfmodel     

A fitted (and valid) ARF model object of class `model`.

options     

An "options" object. The options object belonging to the ARF model is loaded by default.
**Details**

BIC calculates the BIC value using the number of brain voxels (Set by the `model@mask` slot). Without constants the BIC equals:

\[ \text{BIC} = \ln(S(\theta)) + p \times \ln(n) \]

where \( p \) is the number of parameters in the spatial model, \( n \) is the number of brain voxels, and \( S(\theta) \) denotes the minimum of the objective function.

**Value**

Returns an object of class "model" with the slot `model@fit[1]` set to the BIC value.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

RMSEA, model, options

---

`checkSolution`  
Check the integrity of an ARF model

**Description**

Check the integrity of an ARF model for bounded parameters.

**Usage**

`checkSolution(arfmodel, options = loadOptions(arfmodel), dat = readData(.model.avgdatfile(arfmodel)), thres = 6)`

**Arguments**

- `arfmodel`: A model-object (class `model`)
- `options`: An options-object (class `options`)
- `dat`: Average datafile.
- `thres`: Number of digits for equality check.

**Value**

Returns the model object with added warnings if parameters are on a bound.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com
clearWarnings

Clear warnings of a model object.

Description

ClearWarnings clears all warnings of a model object, sets the valid slot to TRUE and saves the model object. It asks for confirmation.

Usage

clearWarnings(arfmodel, resetValid = T)

Arguments

arfmodel: Object of class "model" (see model)
resetValid: If TRUE set the @valid slot to TRUE, else the @valid slot is left unchanged.

Details

Warnings are only cleared when starting the minimization procedure on a model object. After minimization all warnings are added to the warnings slot.

Value

None

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

model

createAllAverages

Create Averages of runs for all conditions.

Description

CreateAllAverages creates averages of runs for all conditions in an ARF experiment. It calls createAverages for all known data-directories within the experiment.

Usage

createAllAverages(experiment = NULL)
createAverages

Arguments

experiment An ARF experiment (object of class "experiment", see experiment).

Details

If an ARF experiment is loaded (loadExp), createAllAverages can be called without passing an argument to createAllAverages().

Value

Returns TRUE when no errors are encountered.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

createAverages, experiment, loadExp

createAverages Create averages of runs for a condition.

Description

CreateAverages creates averages of runs for a condition in an ARF experiment.

Usage

createAverages(arfdat, experiment = NULL)

Arguments

arfdat An object of class "data" (data) from a condition.

experiment An ARF experiment (object of class "experiment", see experiment).

Details

When an arf experiment is loaded using loadExp createAverages can be called with only passing the arfdat object.

Value

Returns the "data" object with the links to the average files in the @avgdatfile, @avgwfile, and @avgtstatfile slots.
Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

data, experiment, loadExp, createAllAverages

createRegs

Create registration directories for runs of an ARF condition.

Description

createRegs creates registration directories for runs of an ARF condition and sets the appropriate paths.

Usage

createRegs(arfdata)

Arguments

arfdata An object of class "data" (data) from a condition.

Value

None

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

data, registration, setRegFiles, setRegParams
Description

The "data" class contains all information on the fMRI data of one condition of a subject. Specifically, it contains filenames of the datafiles (with full paths) and some basic information of the data, the number of runs, and additional files used by ARF. Each instance of a model class object inherits a "data" object. The "data" class also uses some slots of the experiment class.

Details

The "data" class contains information on the file-locations of the fMRI data. The fMRI data (outcomes of a GLM analysis) must be available in the /beta and /weights directories of each condition for each subject. The data can be in the form of beta-values with associated squared standard errors or t-values with the standard errors set to 1. In the latter case the only files that have to be copied are the t-values to the /beta directory, the weight-files are then created automatically. The average files are created after a call to createAverages.

Objects

Objects can be created by calls of the form new("data", ...).

Slots

name: Name of the data object, usually 'subjectname - conditionname'.
fullpath: Full path to the data directory.
betafiles: Vector of filenames (full paths) of the beta files (containing beta estimates).
weightfiles: Vector of filenames (full path) of the weight files (containing variances of the beta estimates).
avgdatfile: Filename of the averaged beta file.
avegweightfile: Filename of the averaged weight file.
avgtstatfile: Filename of the average t-statistics (=avgbetafile/sqrt(avgweightfile)) file.
n: Number of 'brain' voxels (excluding voxels not masked).
mask: Vector containing the mask for the data.
ss: Sums-of-squares of the data.
regdir: Full path to the registration directory.
funcdir: Full path to the functional directory.
dataheader: Nifti header information from the avgdataFile.
rungs: Number of runs (= number of beta files).
version: Object of class "version" (see version).
Methods

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

access .classname.slotname(object)
replace .classname.slotname(object)<-

In this case classname is "data" and slotname is for example "avgstatFile". So to access the "avgstatFile" slot of an r-object "mydata" (of class "data") type .settings.avgstatFile(mydata).

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

model, experiment

Examples

showClass("data")

---

determineStartRect Calculate starting values for an ARF model

Description

determineStartRect calculates starting values for an ARF model by searching for maxima and minima in the data, and estimating the width of these maxima/minima.

Usage

determineStartRect(arfmodel, options = loadOptions(arfmodel))
determineStartRectSimple(arfmodel, options = loadOptions(arfmodel))

Arguments

arfmodel An ARF model object (class model)
options An "options" object (class options), by default the options.Rda from the current model is loaded. The value of options@start.maxfac determines the calculation of the spatial extent values, higher settings will lead to greater extent values.

Details

determineStartRect determines the maxima and minima of the data and at each maximum/minimum searches in each dimension (x,y,z) the point where the maximum/minimum value is half its original value. This creates a box around the maximum/minimum, which is set to zero, to avoid searching the same area twice. The widths of this box are taken as starting values for the width-parameters.
differenceCor

Value

Returns the ARF "model" object, with the starting values in the @startval slot.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

model, options

differenceCor  Calculate if correlations differ.

Description

Calculate which correlations of two correlation matrices are different.

Usage

differenceCor(c1, c2, n1, n2 = n1)

Arguments

c1  Correlation matrix of sample 1.
c2  Correlation matrix of sample 2.
n1  Size of sample 1.
n2  Size of sample 2.

Value

z1  Matrix of Fischer Transformed r's.
z2  Matrix of Fischer Transformed r's.
dif  Difference scores (z1-z2).
z  Z-value of the difference.
pval  P-value of z-diff.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com
euclidDist  

*Calculate Euclidian distances of all regions in a model.*

**Description**

euclidDist calculates Euclidian distances between the center locations of each region in an ARF model. This can be used to assess if regions are close together.

**Usage**

euclidDist(arfmodel, thres = 5, quiet = T)

**Arguments**

- `arfmodel` An arf model object (class `model`)
- `thres` Threshold at which to output region information. If regions are closer than `thres` information is printed if `quiet=FALSE`.
- `quiet` Should region information be printed?

**Details**

euclidDist only uses center location, no information on the extent of a region is used.

**Value**

Returns a lower triangular matrix with Euclidian distances of all regions in the model.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

- `model`

---

**experiment-class**

arf3DS4 "experiment" class

**Description**

The "experiment" class is the backbone of the arf3DS4 package. The class contains information on the names of all relevant directories and files (these are inherited from an "settings" object (see `settings`)). Before running an ARF analysis an object of class "experiment" must be loaded (using `loadExp`). Experiment directories can be created using `makeExpDirs`. To view information on an experiment call `getExp`. 
**experiment-class**

**Objects from the Class**

Objects can be created by calls of the form `new("experiment", ...)`. 

**Slots**

- `path`: Path of the experiment.
- `name`: Name of the experiment.
- `subject.num`: Number of subjects.
- `subject.names`: Vector of length(subject.num) containing subject names.
- `condition.num`: Number of conditions.
- `condition.names`: Vector of length(condition.num) containing condition names.
- `expRda`: Name of the .Rda file containing the experiment information.
- `optionsRda`: Name of the .Rda file containing the options for fitting an ARF model.
- `startRda`: Name of the .Rda file containing the starting values for fitting an ARF model.
- `dataRda`: Name of the .Rda file containing the fMRI data location and information.
- `modelRda`: Name of the .Rda file containing the ARF model information.
- `statsRda`: Name of the .Rda file containing the statistics information of a model (not yet implemented).
- `regRda`: Name of the .Rda file containing the registration information of an fMRI data image.
- `funcRda`: Name of the .Rda file containing the functional information of an fMRI data image.
- `subjectPrefix`: Prefix for the subject names. Defaults to empty.
- `conditionPrefix`: Prefix for the condition names. Defaults to empty.
- `modelPrefix`: Prefix for the model names. Defaults to empty.
- `subjectDir`: Name of the subjects directory.
- `conditionDir`: Name of the conditions directory.
- `dataDir`: Name of the data directory.
- `weightsDir`: Name of the weights directory (within conditions/data).
- `avgDir`: Name of the average data directory (within conditions/data).
- `regDir`: Name of the registration directory (within conditions/data).
- `funcDir`: Name of the functional directories (within subjects and conditions/data).
- `betaDir`: Name of the beta directory (within conditions/data).
- `modelDir`: Name of the model directory.
- `statsDir`: Name of the stats directory (not yet implemented).
- `modelDatDir`: Name of the data directory (within models).
- `avgdatFile`: Name of the averaged data file (fMRI).
- `avgWFile`: Name of the averaged weight file (fMRI).
- `avgstatFile`: Name of the averaged t-values file (fMRI).
- `modelDataFile`: Name of the model prediction file (fMRI).
modelnamesRda: Name of the modelnames.Rda file containing the fitted models.
residualFile: Name of the residuals file (binary).
derivativeFile: Name of the derivatives file (binary).
weightFile: Name of the weight file (binary).
lowresFile: Name of the low resolution structural image.
lowresAvg: Name of the average low resolution structural image.
LogFile: Name of the logFile (not yet implemented).
version: Object of class "version" (see version)

Extends
Class "settings", directly.

Methods
Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

access .classname.slotname(object)
replace .classname.slotname(object)<-

In this case classname is "experiment" and slotname is for example "avgdatfile". So to access the "avgdatfile" slot of an r-object "myexperiment" (of class "experiment") type .experiment.avgdatfile(myexperiment).

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

See Also
settings, makeExpDirs, loadExp, getExp

Examples

showClass("experiment")

extr-repl-fmridata Extract/replace elements of an fmri.data object

Description
Extract or replace elements of an fmri.data object

Details
fMRI data can be accessed using standard R array indexing. The dimensions of the array are determined by the dimensions of the fmri.data object. Both 3D and 4D (time-series) data can be used.
**Value**

Returns an object of class "fmri.data" (in the case of replacement), with the appropriate values replaced) Or returns the appropriate part of the array.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

fmri.data, fmri2array

---

**fitConnectivity**

*Estimate connectivity of an ARF model*

**Description**

fitConnectivity estimates the functional connectivity between activated regions in an ARF model based on trial-to-trial amplitude variation. It uses the single-trial data (extracted from the raw time-series using makeSingleTrialEvents), to estimate the trial-by-trial amplitudes of each region in the ARF model.

**Usage**

```r
fitConnectivity(arfmodel, funcfilename="single_events.nii.gz")
```

**Arguments**

- **arfmodel** An ARF model object (class `model`)
- **funcfilename** Filename of the volume containing the single-trial data.

**Details**

Each volume of the file specified in funcfilename should contain estimates of a single-trial (see makeSingleTrialEvents).

**Value**

Returns an object of class `arfcorrelation` including the region by time matrix of amplitude time-series and the estimated correlations.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

model, makeSingleTrialEvents, arfcorrelation, roiConnectivity
**fitModel**

Fit an ARF model.

**Description**

fitModel performs the fit-procedure of an ARF model to the data. It uses the values in the `options` object to modify the behavior of the fit-procedure. By default `optim` (L-BFGS-B) is used for minimization. fitModel is a wrapper for `fitModelOptim` and `fitSimpleModelOptim`.

**Usage**

```r
glmm(..., data = ..., weights = ..., start = ..., method = ..., control = ..., n那次 = ..., verbose = ...)
```

**Arguments**

- `arfmodel`: An arf model object (class `model`).
- `options`: An "options" object (class `options`). The options@start.method slot is used to determine which method to use for starting values. The arfmodel@modeltype slot is used to determine whether to fit a full Gaussian model (default) or a simple Gaussian model (with fixed width parameters).
- `dat`: An object of class "fmri.data" (`fmri.data` containing the averaged beta-values).
- `weights`: An object of class "fmri.data" (`fmri.data` containing the averaged weights of the beta-values).
- `printlevel`: Number indicating the level of output of the minimization procedure.
- `try.silen`: Should errors be printed when they occur or only passed to the warnings slot.

**Details**

fitModel calls one of the following functions based on the "options" object: `fitModelOptim`, `fitSimpleModelOptim`. These functions can also be called directly. Models can be created by calling `newModel`.

**Value**

Returns an object of class "model" with the appropriate slots.
fitRange

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

See Also
model, options, fmri.data, newModel

fitRange

Fit a range of models.

Description
fitRange performs model fits for a range of models.

Usage
fitRange(subject, condition, range, options=new('options'),
  modelprefix='searchmodel', modeltype=c('gauss','simple'),
  experiment=NULL)

Arguments
subject Name of the subject.
condition Name of the condition.
range Vector containing the different numbers of regions for the spatial model.
options An options object.
modelprefix Prefix to append before the model name.
modeltype Fit a full Gaussian or simple Gaussian model.
experiment Optional experiment object (by default uses .experiment).

Value
Returns an object of class sequence holding information on the valid models (minimum, BIC value, number of regions).

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

See Also
model, options, experiment
The arf3DS4 "fmri.data" class contains the header information and data of an ANALYZE/NIFTI functional volume. It is essentially a nifti.header object with an added "datavec" slot.

Objects

Objects can be created by calls of the form new("fmri.data", ...).

Slots

datavec: Vector containing the actual fMRI data.
sizeof_hdr: <nifti> Size of the header file.
data_type: <nifti> Type of data (representation).
db_name: <nifti> db_name.
extents: <nifti> extents
session_error: <nifti> session error.
regular: <nifti> regular
dim_info: <nifti> MRI slice ordering.
dims: <nifti> Data array dimensions.
intent_p1: <nifti> 1st intent parameter.
intent_p2: <nifti> 2nd intent parameter.
intent_p3: <nifti> 3rd intent parameter.
intent_code: <nifti> Intent code.
datatype: <nifti> Data type.
bitpix: <nifti> Number of bits per voxel.
slice_start: <nifti> First slice index.
pixdim: <nifti> Grid spacing (single voxel dimensions).
vox_offset: <nifti> Offset in the .nii file.
scl_slope: <nifti> Data scaling, slope.
scl_inter: <nifti> Data scaling, offset.
slice_end: <nifti> Last slice index.
slice_code: <nifti> Slice timing order.
xyzt_units: <nifti> Units of single voxel dimensions.
cal_max: <nifti> Maximum display intensity.
cal_min: <nifti> Minimum display intensity.
slice_duration: <nifti> Time for one slice (TR).
toffset: <nifti> Shift in time.
glmax: <nifti> glmax.
glmin: <nifti> glmin.
descrip: <nifti> Description of file.
aux_file: <nifti> An auxilliary filename.
qform_code: <nifti> q_form code.
sform_code: <nifti> s_form code.
quatern_b: <nifti> quaternion b parameter.
quatern_c: <nifti> quaternion c parameter.
quatern_d: <nifti> quaternion d parameter.
qoffset_x: <nifti> q offset x parameter.
qoffset_y: <nifti> q offset y parameter.
qoffset_z: <nifti> q offset z parameter.
srow_x: <nifti> 1st row affine transformation.
srow_y: <nifti> 2nd row affine transformation.
srow_z: <nifti> 3rd row affine transformation.
intent_name: <nifti> Meaning of data.
magic: <nifti> nifti magicstring.
data.type: <arf3DS4> Data type.
data.signed: <arf3DS4> Signed data.
fullpath: <arf3DS4> Full path of datafile.
filename: <arf3DS4> Filename.
filetype: <arf3DS4> Type of file.
extension: <arf3DS4> File extension.
gzipped: <arf3DS4> Is the file gzipped?
endian: <arf3DS4> Endianness of file.
version: Object of class "version" (see version)

Extends

Class "nifti.header", directly. Class "nifti.fileinfo", by class "nifti.header".

Methods

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

access .classname.slotname(object)
replace .classname.slotname(object)<-
plot signature(x = "fmri.data", y = "missing"); ...
show signature(object = "fmri.data"); ...
summary signature(object = "fmri.data"); ...

In this case classname is "fmri.data" and slotname is for example "dims". So to access the "dims" slot of an r-object "myfmridata" (of class "fmri.data") type .fmri.data.dims(myfmridata).
**Description**

Convert the data vector of an fmri.data object to an array.

**Usage**

```r
fmri2array(fmrdat)
```

**Arguments**

- `fmrdat`  An object of class `fmri.data`.

**Value**

Returns an array containing the fMRI data (@datavec-slot) with dimensions set by the fmri.data object.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

- `fmri.data`
The arf3DS4 "functional" class contains information on the functional volume where the fMRI beta-files were estimated on. It links the data of the different runs with the raw time-series files, primarily for connectivity analysis. To link runs to raw time-series call setFuncTimings.

Objects

Objects can be created by calls of the form new("functional", ...).

Slots

fullpath: Full path of the functional volume directory.
linkedfiles: Path to linked beta-file.
timings: Vector of stimulus timings.
version: Object of class "version" (see version)

Methods

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

access .classname.slotname(object)
replace .classname.slotname(object)<-

In this case classname is "functional" and slotname is for example "timings". So to access the "timings" slot of an r-object "myfunctional" (of class "functional") type functional.timings(myfunctional).

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

Examples

showClass("functional")
getAtlasLabels

*Find anatomical atlas labels.*

**Description**

getAtlasLabels finds anatomical atlas labels for a set of coordinates (either native ARF space or standard MNI space). It requires a working copy of FSL.

**Usage**

getAtlasLabels(coordinates, registration, coortype = c("arf", "mni"), atlas = c("both", "Talairach", "HarvardOxford"), ...)

**Arguments**

- **coordinates**: Matrix with three columns (x,y,z) containing coordinates of a region. Multiple regions are in the rows of the matrix.
- **registration**: A "registration" object belonging to the data the coordinates are from (see registration).
- **coortype**: In what coordinate space are the coordinates (ARF native or MNI space).
- **atlas**: For which atlases should the labels be returned? Currently only 'talairach' and 'harvardOxford' are available.

**Details**

getAtlasLabels only works with a working copy of a full FSL installation (>=4.1).

**Value**

Returns a list of length nrow(coordinates) with a talairach and/or a harvard slot containing the labels.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**References**

See http://www.fmrib.ox.ac.uk/fsl/ on how to obtain FSL and licences. See http://www.fmrib.ox.ac.uk/fsl/data/atlas-descriptions.html on information on the atlases.

**See Also**

registration, model, setRegFiles
getExp

Access experiment variables.

Description
Access the experiment (if loaded).

Usage
getExp()

Value
Returns an object of class experiment.

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

See Also
experiment, loadExp

getModelAtlas

Get Atlas Labels for an ARF model.

Description
getModelAtlas fetches atlas labels for the center coordinates of all regions in a model.

Usage
getModelAtlas(arfmodel, regrun = 1, saveastext = F)

Arguments
arfmodel Object of class "model" (see model).
regrun Which run is used for the coordinate conversion?
saveastext Save a text file in the working directory with atlas information?

Details
getModelAtlas only works with a working copy of a full FSL installation (>=4.1). It calls getAtlasLabels.
Value

Returns a list of length nrow(coordinates) with a talairach and/or a harvard slot containing the labels.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

References

See http://www.fmrib.ox.ac.uk/fsl/ on how to obtain FSL and licences. See http://www.fmrib.ox.ac.uk/fsl/data/atlas-descriptions.html on information on the atlases.

See Also

model, getAtlasLabels

loadData  

Load an arf3DS4 "data" object.

Description

loadData loads a "data" object for a given subject and condition.

Usage

loadData(subject, condition, experiment = NULL)

Arguments

subject Name of the subject.
condition Name of the condition.
experiment Optional object of class "experiment" (see experiment)

Details

loadData uses the loaded experiment as a default (see loadExp) on how to load experiments.

Value

Returns an object of class "data" (see data).

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

data, experiment, loadExp
loadExp

Load an arf3DS4 experiment

Description

loadExp loads an arf3DS4 experiment.

Usage

loadExp(path = getwd(), method = c("fast", "set", "rda"))

Arguments

path        Path to the experiment directory.
method      Which method is used for loading.

Details

If method = 'fast' (default) experiment.Rda is loaded without checks directly. If method = 'set' the experiment is entirely build up by searching the directories of the experiment for subjects/conditions and models. Use this method if the experiment has changed (added subjects, moved to another directory etc.). If method = 'rda' the experiment is checked using the data in the experiment.Rda (except for the root-path). Use this method if the experiment has been moved (but the subjects/conditions have not changed). loadExp creates an object named .experiment in the .arfInternal environment, this is to facilitate calls to functions which have an 'experiment' object as input. All ARF functions get the .experiment object from .arfInternal.

Value

Returns an object of class experiment invisibly.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

experiment
loadFunc  
*Load functional data for a single run of a condition.*

**Description**

Load functional data (linking the run to the raw time-series) for a single run of a condition. This information is used by `makeSingleTrialEvents` to estimate single-trial data.

**Usage**

```r
loadFunc(subject, condition, run, experiment = NULL)
```

**Arguments**

- `subject`  
  Subject name.
- `condition`  
  Condition name.
- `run`  
  Which run (can be numeric, or a directory name).
- `experiment`  
  Optional experiment object.

**Value**

Returns an object of class `functional`.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

`functional`, `experiment`, `setFuncTimings`

---

loadModel  
*Load an ARF model*

**Description**

loadModel loads an ARF model based on the `modelname`, `subject` name and `condition` name.

**Usage**

```r
loadModel(modelname, subject = NA, condition, experiment = NULL)
```
**loadOptions**

**Arguments**

- **modelname**
  - Name of the model to be loaded.
- **subject**
  - Name of the subject.
- **condition**
  - Name of the condition.
- **experiment**
  - Optional object of class `experiment`.

**Details**

loadModel can also take as input an object of class `mnames` with as a second (subject) argument a numerical indicator of which model to load.

**Value**

Returns an object of class `model`.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

`model`, `experiment`, `mnames`

---

### loadOptions

**Save/load ARF Options**

**Description**

Load or save an "options" object belonging to an ARF model.

**Usage**

```r
loadOptions(arfmodel)
saveOptions(options, arfmodel)
```

**Arguments**

- **arfmodel**
  - Object of class "model" (see `model`)
- **options**
  - Object of class "options" (see `options`)

**Value**

loadOptions returns an object of class `options`.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com
See Also

model, options

loadRda

Load an .Rda file into a specified object.

Description

loadRda loads a standard .Rda file, and returns this object (instead of loading it directly in the R workspace).

Usage

loadRda(file)

Arguments

file Filename

Details

loadRda assumes that only one object was saved. It cannot load an entire workspace.

Value

Returns the object that was saved in the .Rda file.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

loadReg

Load a registration object.

Description

Load a registration object for a single run of a condition.

Usage

loadReg(subject, condition, run, experiment = NULL)
**loadStart**

**Arguments**

subject Subject.
condition Condition.
run Which run (can be numeric, or a directory name)
experiment Optional experiment object.

**Value**

Returns an object of class registration.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

registration, experiment

---

**loadStart**

*Save/load ARF starting values*

**Description**

Load the starting values of an ARF model from the .Rda file

**Usage**

loadStart(arfmodel)
saveStart(startval, arfmodel)

**Arguments**

arfmodel Object of class "model" (see model)
startval Vector of starting values.

**Value**

loadStart returns a vector with starting values.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

model
makeDerivs

Calculate analytical first-order derivatives of parameters in an ARF model.

Description

makeDerivs calculates analytical first-order derivatives of the parameters in an ARF model. It writes the derivative matrix to a binary file.

Usage

makeDerivs(arfmodel, method=c('viaR', 'direct'))

Arguments

arfmodel An arf model object (see model).
method Specification of derivative saving method. 'viaR' is the default.

Value

Returns TRUE when successful.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

model

makeExpDirs

Make Experiment directories.

Description

makeExpDirs makes an ARF experiment directory structure.

Usage

makeExpDirs(path = getwd(), name = "default_experiment", subjectInd = 1, conditionInd = 1, settings = new("settings"))
**Arguments**

- `path`  
  Path in which to create the experiment directory.
- `name`  
  Name of the experiment (=name of the directory).
- `subjectind`  
  Vector of subject names for which to create directories.
- `conditionind`  
  Vector of condition names for which to create directories.
- `settings`  
  An optional "settings" object (see `settings`).

**Details**

`makeExpDirs` only makes the directory structure, data files must be copied to the directories manually (see `data`). After copying a call to `loadExp` in mode 'set', will set the experiment.Rda.

**Value**

Returns an "experiment" object (`experiment`).

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

- `settings`, `loadExp`, `experiment`

---

**Description**

Make low resolution structural image from high resolution T1 image. The function requires that registration files (`setRegFiles`) and parameters (`setRegParams`) are set. It also requires a working copy of FSL.

**Usage**

```r
makeLowResStruct(arfdata, experiment = NULL)
makeLowResStructAvg(arfmodel, experiment = NULL)
```

**Arguments**

- `arfdata`  
  An arf "data" object (see `data`).
- `arfmodel`  
  An arf "model" object (see `model`).
- `experiment`  
  An arf "experiment" object (see `experiment`).
Details

makeLowResStructAvg makes use of the images created by makeLowResStruct so this must be called first. makeLowResStruct has to be run only once for each condition.

Value

None.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

References

See [http://www.fmrib.ox.ac.uk/fsl/](http://www.fmrib.ox.ac.uk/fsl/) on how to obtain FSL and licences.

See Also

model, experiment, data, setRegFiles, setRegParams

---

**makeROImask**

*Mask an fMRI image*

**Description**

MakeROImask masks an fMRI image based on the non-zero elements of a maskfile (of class `fmri.data`).

**Usage**

```r
makeROImask(fmridata, maskdata)
```

**Arguments**

- `fmridata`: The data to be masked (class `fmri.data`).
- `maskdata`: The mask to be applied (voxels with non-zero-values are masked), can be of class `fmri.data` or a numeric vector.

**Value**

Returns a masked object of class `fmri.data`.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

`fmri.data`
**makeSingleTrialEvents**

*Estimate single-trial fMRI activity.*

**Description**

`makeSingleTrialEvents` calculates single-trial fMRI activity using a double gamma HRF.

**Usage**

```r
makeSingleTrialEvents(subject, condition, sefilename='single_events',
                      hrf.control=list(a1=6, a2=12, b1=0.9, b2=0.9, ce=0.35), experiment=NULL)
```

**Arguments**

- **subject**: Name of the subject.
- **condition**: Name of the condition.
- **sefilename**: Filename of the volume where the single-trial data is stored (stores a 4D file with a volume for each event).
- **hrf.control**: A list containing the parameters for the double gamma HRF.
- **experiment**: An optional object of class `experiment`.

**Details**

Files are saved in the `data/functional` directory using the filename specified in `sefilename`. The function also returns the single-trial data.

**Value**

Returns a 4D nifti file (of class `fmri.data`) containing the single-trial data.

**Warnings**

Before `makeSingleTrialEvents` can be executed, the data of each run within a condition must be linked to the raw time-series. This is done via object of class `functional` by calling `setFuncTimings`.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

`model`, `fmri.data`, `experiment`, `functional`, `setFuncTimings`
mcpCorrect  

Calculate Multiple Comparison Corrections

Description

mcpCorrect calculates three types of multiple comparison corrections: 'uncorrected', 'bonferroni', and 'False Discovery Rate (FDR)'. mcpCorrect assumes the data are t-values.

Usage

mcpCorrect(fmridata, type = c("uncorrected", "bonferroni", "FDR"),
alpha = 0.05, q = 0.05, cv = 1, df = 100, sig.steps = 1, adj.n = T)

Arguments

fmridata  An object of class "fmri.data" (see fmri.data).
type  Type of correction ('uncorrected', 'bonferroni', 'FDR')
alpha  Nominal alpha level.
q  q parameter for FDR.
cv  Cv parameter for FDR.
df  Degrees of freedom of the t-values.
sig.steps  Number of steps to divide p-values in (for visualization).
adj.n  Use only brain voxels when correcting?

Value

Returns two object of class "fmri.data", one with suprathreshold voxels masked, one with only significant voxels used for overlay images.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

fmri.data
minBIC

Calculate BIC values for multiple ARF models.

Description

Calculate and show the BIC, minimum and number of regions of all models in a condition.

Usage

minBIC(subject, condition)

Arguments

subject Name of the subject.
condition Name of the condition.

Value

Returns an object of class sequence containing fit-information of all models.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

fmri.data, sequence

mnames-class

arf3DS4 "mnames" class

Description

The arf3DS4 "mnames" class contains information on the models in a subject/condition. It can be used to quickly load models without having to specify the experiment, subject and condition each time.

Objects

Objects can be created by calls of the form new("mnames", ...).

Slots

experiment: Object of class "experiment" (see experiment).
subject: Subject name.
condition: Condition name.
mnames: Vector of modelnames within a subject/condition.
Methods

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

```
access .classname.slotname(object)
replace .classname.slotname(object)<-
show signature(object = "mnames"): ... 
```

In this case classname is "mnames" and slotname is for example "subject". So to access the "subject" slot of an r-object "mymnames" (of class "mnames") type .mnames.subject(mymnames).

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

experiment

Examples

```
showClass("mnames")
```

---

### model-class

**arf3DS4 "model" class**

---

**Description**

The arf3DS4 model class contains all information on a (fitted) model. It extends the "data" class directly, since it uses the files specified there to fit the model. The "model" class contains information on file locations and filenames. More importantly it contains information on the fitted model: parameter estimates, parameter (co)variance matrix, statistical test information, fit indices, convergence information, warnings, etc.

**Objects**

Objects can be created by calls of the form `new("model", ...)`.

**Slots**

- `modelName`: Name of the (fitted) model.
- `modelpath`: Full path to the model directory.
- `modelDatapath`: Full path to the model/data directory.
- `residualFile`: Name of the file containing model residuals.
- `derivativeFile`: Name of the file containing first order derivatives.
- `weightFile`: Name of the binary file containing weights.
- `modelDataFile`: Name of the file containing model predictions.
fullmodelDataFile: Full name of the file containing model predictions.
modelName: Name of the model.Rda file containing the model object.
optionsFile: Name of the options.Rda file (used to specify options, see options).
startFile: Name of the start.Rda file.
convergence: Convergence information.
iterates: Number of iterates.
minimum: Value of the objective function at the minimum.
estimates: Parameter Estimates.
gradients: Gradient of the parameters at the minimum.
hessian: Hessian matrix of parameters.
params: Number of parameters.
modeltype: Type of model fitted.
sandwichmethod: Method used to calculate Sandwich estimates.
varcov: Variance/covariance matrix of the parameters.
warnings: Warnings encountered during processing.
fit: Fit of the model (BIC and RMSEA).
wald: Wald class object (containing model statistics, see wald).
regions: Number of regions in the spatial model.
startval: Starting values of the minimization procedure.
proctime: Processing time of minimization and sandwich procedures.
valid: Is the model valid?
name: Name of the data object, usually 'subjectname - conditionname'.
fullpath: Full path to the data directory.
betafiles: Vector of filenames (full paths) of the beta files (containing beta estimates).
weightfiles: Vector of filenames (full paths) of the weight files (containing variances of the beta estimates).
avgdatfile: Filename of the averaged beta file.
avgWfile: Filename of the averaged weight file.
avgtstatFile: Filename of the average t-statistics (=avgbetafile/sqrt(avgweightfile)) file.
n: Number of 'brain' voxels (i.e. the number of voxels excluding those where @mask is zero).
mask: Vector containing the mask for the data.
ss: Sums-of-squares of the data.
regDir: Fullpath to the registration directory.
funcDir: Fullpath to the functional directory.
dataHeader: Nifti header information from the avgdataFile.
runs: Number of runs (= number of beta files).
version: Object of class "version" (see version)
**newModel**

Create a new ARF model

**Description**

`newModel` creates a new model of with a given number of regions in the spatial model within a subject/condition, using the options specified. It also creates a new directory in the `model` directory of the condition.

**Usage**

```r
newModel(modelname = "defaultmodel", regions = 1, subject = "", condition = "", type = c("gauss", "simple"), options = new("options"), overwrite = T, experiment = NULL)
```
Arguments

modelname  Name of the model.
regions    Number of regions in the spatial model.
subject    Name of the subject.
condition  Name of the condition.
type       Type of model to fit (full Gaussian ('gauss'), simple Gaussian ('simple')).
options    Options object (options).
overwrite  Overwrite the model if a model with the same name already exists?
experiment Experiment object (as default uses .experiment).

Details

Calls to newModel save an instance of a file named model.Rda in the appropriate directory and return the model object.

Value

Returns an object of class "model" with the appropriate slots.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

model, options, experiment

nifti.fileinfo-class arf3DS4 "nifti.fileinfo" class

Description

The arf3DS4 "nifti.fileinfo" class contains the basic file information needed to read/write an ANALYZE/NIFTI header.

Objects

Objects can be created by calls of the form new("nifti.fileinfo", ...).
Slots

fullpath: Full path of fMRI datafile.
filename: Filename of the fMRI datafile.
filetype: Type of the fMRI datafile.
extension: File extension.
gzipped: Is the file gzipped?
endian: Endianness of file.
version: Object of class "version" (see version)

Methods

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

access .classname.slotname(object)
replace .classname.slotname(object)<-

In this case classname is "nifti.fileinfo" and slotname is for example "extension". So to access the "extension" slot of an r-object "myniftifileinfo" (of class "nifti.fileinfo") type .nifti.fileinfo.extension(myniftifileinfo).

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

References

See the Nifti homepage (http://nifti.nimh.nih.gov/) for more information on the nifti filetype.

See Also

nifti.header, fmri.data

Examples

showClass("nifti.fileinfo")

nifti.header-class

Description

The arf3DS4 "nifti.header" class contains the header information of an ANALYZE/NIFTI functional volume.

Objects

Objects can be created by calls of the form new("nifti.header", ...).
Datasets

- **sizeof_hdr**: Size of the header file.
- **data_type**: Type of data (representation).
- **db_name**: db name.
- **extents**: extents
- **session_error**: session error.
- **regular**: regular
- **dim_info**: MRI slice ordering.
- **dims**: Data array dimensions.
- **intent_p1**: 1st intent parameter.
- **intent_p2**: 2nd intent parameter.
- **intent_p3**: 3rd intent parameter.
- **intent_code**: Intent code.
- **datatype**: Data type.
- **bitpix**: Number of bits per voxel.
- **slice_start**: First slice index.
- **pixdim**: Grid spacing (single voxel dimensions).
- **vox_offset**: Offset in the .nii file.
- **scl_slope**: Data scaling, slope.
- **scl_inter**: Data scaling, offset.
- **slice_end**: Last slice index.
- **slice_code**: Slice timing order.
- **xyzt_units**: Units of single voxel dimensions.
- **cal_max**: Maximum display intensity.
- **cal_min**: Minimum display intensity.
- **slice_duration**: Time for one slice (TR).
- **toffset**: Shift in time.
- **glmax**: glmax.
- **glmin**: glmin.
- **descrip**: Description of file.
- **aux_file**: An auxiliary filename.
- **qform_code**: q_form code.
- **sform_code**: s_form code.
- **quatern_b**: Quaternion b parameter.
- **quatern_c**: Quaternion c parameter.
- **quatern_d**: Quaternion d parameter.
- **qoffset_x**: q offset x parameter.
qoffset_y: <nifti> q offset y parameter.
quoffset_z: <nifti> q offset z parameter.
srow_x: <nifti> 1st row affine transformation.
srow_y: <nifti> 2nd row affine transformation.
srow_z: <nifti> 3rd row affine transformation.
intent_name: <nifti> Meaning of data.
magic: <nifti> nifti magicstring.
data.type: <arf3DS4> Data type.
data.signed: <arf3DS4> Signed data.
fullpath: <arf3DS4> Full path of datafile.
filename: <arf3DS4> Filename.
filetype: <arf3DS4> Type of file.
extension: <arf3DS4> File extension.
gzipped: <arf3DS4> Is the file gzipped?
edian: <arf3DS4> Endianness of file.
version: Object of class "version" (see version)

Extends
Class "nifti.fileinfo", directly.

Methods
Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

access .classname.slotname(object)
replace .classname.slotname(object)<-

In this case classname is "nifti.header" and slotname is for example "dims". So to access the "dims" slot of an r-object "myniftiheader" (of class "nifti.header") type .nifti.header.dims(myniftiheader).

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

References
See the Nifti homepage (http://nifti.nimh.nih.gov/) for more information on the nifti filetype.

See Also
fmri.data,nifti.fileinfo

Examples
showClass("nifti.header")
The `arf3DS4` "options" class contains the options of the minimization procedure (including starting value calculation, and (co)variance estimation). Mostly used by ARF "model" objects (see `model`).

Objects

Objects can be created by calls of the form `new("options", ...)`. 

Slots

- `nls.gradtol`: `<nls>` Gradient tolerance (not used).
- `nls.steptol`: `<nls>` Step tolerance (not used).
- `opt.method`: `<optim>` Optim method (L-BFGS-B).
- `opt.lower`: `<optim>` Lowerbound of parameters (for L-BFGS-B).
- `opt.upper`: `<optim>` Upperbound of parameters (for L-BFGS-B).
- `min.analyticalgrad`: Use analytical gradient in minimization?
- `min.iterlim`: Iteration limit.
- `min.boundlim`: Persistent Boundary iteration limit.
- `min.routine`: Minimization routine (optim).
- `start.method`: Method of starting value calculation ('use', 'load', or 'rect')
- `start.maxfac`: Width of smoothing kernel used in starting value calculation.
- `start.vector`: Starting value template (used for default values).
- `chk.method`: Check method for parameters of minimized solution.
- `chk.range`: Range of values to check parameters of minimized solution.
- `sw.type`: Use diagonal residual matrix or full residual matrix in sandwich (co)variance calculation.
- `output.mode`: Vector with output types ('none' or 'progress').
- `version`: Object of class "version" (see `version`)

Methods

Standard `arf3DS4` accessor and replacement functions can be used to access and replace slots.

```r
access .classname.slotname(object)
replace .classname.slotname(object)<-
```

In this case `classname` is "options" and `slotname` is for example "sw.type". So to access the "sw.type" slot of an r-object "myoptions" (of class "options") type `options.sw.type(myoptions)`. 

---

**Description**

The `arf3DS4` "options" class contains the options of the minimization procedure (including starting value calculation, and (co)variance estimation). Mostly used by ARF "model" objects (see `model`). 

**Objects**

Objects can be created by calls of the form `new("options", ...)`. 

**Slots**

- `nls.gradtol`: `<nls>` Gradient tolerance (not used).
- `nls.steptol`: `<nls>` Step tolerance (not used).
- `opt.method`: `<optim>` Optim method (L-BFGS-B).
- `opt.lower`: `<optim>` Lowerbound of parameters (for L-BFGS-B).
- `opt.upper`: `<optim>` Upperbound of parameters (for L-BFGS-B).
- `min.analyticalgrad`: Use analytical gradient in minimization?
- `min.iterlim`: Iteration limit.
- `min.boundlim`: Persistent Boundary iteration limit.
- `min.routine`: Minimization routine (optim).
- `start.method`: Method of starting value calculation ('use', 'load', or 'rect')
- `start.maxfac`: Width of smoothing kernel used in starting value calculation.
- `start.vector`: Starting value template (used for default values).
- `chk.method`: Check method for parameters of minimized solution.
- `chk.range`: Range of values to check parameters of minimized solution.
- `sw.type`: Use diagonal residual matrix or full residual matrix in sandwich (co)variance calculation.
- `output.mode`: Vector with output types ('none' or 'progress').
- `version`: Object of class "version" (see `version`)

**Methods**

Standard `arf3DS4` accessor and replacement functions can be used to access and replace slots.
Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

See Also
model

Examples
showClass("options")

processModel
Process an ARF model.

Description
Run an entire processing pipeline for a given ARF model (minimization and statistics).

Usage
processModel(arfmodel, options = loadOptions(arfmodel),
  dat = readData(.model.avgdatfile(arfmodel)),
  weights = readData(.model.avgWfile(arfmodel)),
  pr = T, printlevel = 0, try.silen = T)

Arguments
arfmodel An arf model object (class model).
options An "options" object (class options). The options@start.method slot is used to
determine which method to use for starting values. The arfmodel@modeltype
slot is used to determine whether to fit a full Gaussian model (default) or a
simple Gaussian model.
dat An object of class "fmri.data" (fmri.data containing the averaged beta-values.)
weights An object of class "fmri.data" (fmri.data containing the averaged weights of
the beta-values.)
pr Print output to the screen, or perform silently.
printlevel Number indicating the level of output of the minimization procedure.
try.silen Should errors be printed when they occur or only passed to the @warnings slot.

Details
processModel calls fitModel, varcov and wald. The model is automatically saved.

Value
Returns an object of class "model" with the appropriate slots.
**processSeed**

**Author(s)**
Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**
model, options, fmri.data, newModel, fitModel, varcov, wald

---

**processSeed**

Process a model using a seed number of regions

---

**Description**

processSeed runs a sequence of models with different starting values to obtain a stable solution. It starts with a simple model with rectangular starts, subsequently it runs a full model given the estimates of the simple model. It then prunes the full model, discarding the regions with bad gradients and/or bounded parameters.

**Usage**

```r
processSeed(modelname='defaultmodel', seedreg, subject='',
condition='', startmethod=c('default','simple'), grad=NULL,
bound=NULL, pval=NULL, options=new('options'), pr=T,
printlevel=0, try.silen=T, overwrite=T, experiment=NULL)
```

**Arguments**

- `modelname` Name to give the pruned models in the sequence.
- `seedreg` Number of regions to fit at sequence start (seed number).
- `subject` Name of the subject.
- `condition` Name of the condition.
- `startmethod` Use simple model as start or the default (rectangular method).
- `grad` Gradient check limits, absolute higher values are pruned.
- `bound` Threshold to check if a parameter is on a bound. When NULL, check is suppressed.
- `pval` Prune regions with non-significant Wald statistics higher than `pval`.
- `options` Options object.
- `pr` Output to screen?
- `printlevel` Number indicating the level of output of the minimization procedure.
- `try.silen` Should errors be printed when they occur or only passed to the `warnings` slot.
- `overwrite` Overwrite the model if a model with the same name already exists?
- `experiment` Optional experiment object.
pruneModel

Value

Returns the best model (class `model`)

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

`model`, `newModel`, `fitModel`, `pruneModel`, `options`, `experiment`

---

**pruneModel**

*Prune a model based on invalid gradients and bounded parameters.*

Description

Prune model prunes an ARF model according to the gradients and bounds.

Usage

```r
pruneModel(arfmodel, modelName='defaultmodel', subject=''
condition='', grad=NULL, bound=NULL, pval=NULL,
options=new('options'), overwrite=T, experiment=NULL)
```

Arguments

- `arfmodel` An arf model object (class `model`).
- `modelName` Name for the new models (pruneModel makes newmodelnames).
- `subject` Name of the subject.
- `condition` Name of the condition.
- `grad` Absolute threshold for gradient check.
- `bound` Number of digits used in equality check on the bound.
- `pval` Prune non-significant regions with p-values higher than pval.
- `options` An "options" object (class `options`).
- `overwrite` Overwrite existing objects?
- `experiment` Optional experiment object.

Value

Returns an object of class "model" with the appropriate slots.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com
readData

See Also

model, options, experiment, processSeed

readData

read in a NIFTI/ANALYZE file.

Description

readData reads in a NIFTI/ANALYZE file.

Usage

readData(filename)

Arguments

filename Name of the file to be read in.

Value

Returns an object of class fmri.data.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

References

See the Nifti homepage (http://nifti.nimh.nih.gov/) for more information on the nifti filetype.

See Also

fmri.data
readDerivs Read (binary) derivative file

Description
Read in derivatives of an ARF model.

Usage
readDerivs(arfmodel)

Arguments
arfmodel A valid ARF model object.

Value
Returns an object of class "matrix" containing the derivatives (voxels x parameters).

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

See Also
model.

registration-class arf3DS4 "registration" class

Description
The arf3DS4 registration class contains information on the conversion matrices for brain registration purposes. It is used to convert ARF models to standard space (MNI152), and to structural (T1) images. It is also used to look up the location of activated regions in standard atlases (Harvard-Oxford and Talairach). The "registration" class is optional, for it to work properly a functioning installation of FSL is required.

Objects
Objects can be created by calls of the form new("registration", ...).
Slots

- **fulpath**: Full path of registration directory.
- **filename**: Registration filename.
- **linkedfile**: Path to linked beta-file.
- **examp2high**: ARF to high resolution affine transformation.
- **high2stand**: High resolution to standard space affine transformation.
- **examp2stand**: ARF to standard space affine transformation.
- **example**: Nifti file in arf space.
- **highres**: T1 weighted structural image.
- **standard**: Standard space image (MNI152).
- **Dex**: ARF voxel to ARF mm.
- **Dhi**: highres mm to highres voxels.
- **Dst**: standard mm to standard voxels.
- **SXhi**: x-axis flip.
- **Aex2hi**: ARF mm to highres mm.
- **Ahi2st**: highres mm to standard mm.
- **OXst**: standard origin offset.
- **version**: Object of class "version" (see version)

Methods

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

- **access** `.classname.slotname(object)`
- **replace** `.classname.slotname(object)<-`

In this case classname is "registration" and slotname is for example "highres". So to access the "highres" slot of an r-object "myregistration" (of class "registration") type `.registration.highres(myregistration)`.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

References

See [http://www.fmrib.ox.ac.uk/fsl/](http://www.fmrib.ox.ac.uk/fsl/) on how to obtain FSL and licences. See [http://www.fmrib.ox.ac.uk/fsl/data/atlas-descriptions.html](http://www.fmrib.ox.ac.uk/fsl/data/atlas-descriptions.html) on information on the atlases.

Examples

`showClass("registration")`
RMSEA

Calculate Root Mean Square Error.

Description

Calculate RMSEA fit value.

Usage

RMSEA(arfmodel, options = loadOptions(arfmodel))

Arguments

arfmodel A fitted (and valid) arf-model object of class "model".
options An "options" object. The options object belonging to the ARF model is loaded by default.

Details

RMSEA calculates the RMSEA value using the number of brain voxels (Set by the model@mask slot).

Value

Returns an object of class "model" with the slot model@fit[2] set to the RMSEA value.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

BIC, model, options

roiConnectivity

calculate Connectivity with spatial averaging

Description

Calculates functional connectivity based on the spatial average of the voxels within the isoContours of the estimated model.

Usage

roiConnectivity(arfmodel, roidata=setIsoContour(arfmodel, 95), funcfilename='single_events.nii.gz', type=c('avg','ev'), evmodel=c('spatial','spatiotemporal','eigenvariate'))
**saveFunc**

**Arguments**

- `arfmodel`  An ARF model object (class `model`).
- `roidata`  `fmri.data` object containing ROIs for each region (must contain a volume for each region), can be obtained via `setIsoContour`.
- `funcfilename`  Filename of the volume to extract the single-trial data.
- `type`  Average using no weighing ("avg") or by weighing with the first spatial eigenvector ("ev").
- `evmodel`  Which data is used to calculate the first spatial eigenvector: 'spatial', 'spatiotemporal', or 'eigenvariate'.

**Value**

Returns an object of class `arfcorrelation` including the region by time matrix of amplitude time-series and the estimated correlations.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

- `model`, `arfcorrelation`, `fitConnectivity`, `setIsoContour`

```
saveFunc(functional)  # Save functional object.
```

**Description**

Save an object of class "functional".

**Usage**

```
saveFunc(functional)
```

**Arguments**

- `functional`  Object of class `functional`

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

- `functional`, `loadFunc`
saveModel

Save a model object

Description
Save a model object to the model.Rda file.

Usage
saveModel(arfmodel)

Arguments
arfmodel An object of class model.

Value
None

Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

See Also
model

saveModelBin

Save ARF model estimates to a NIFTI file.

Description
Save ARF model estimates to a NIFTI file. It has several options regarding which regions of the spatial model are saved.

Usage
saveModelBin(arfmodel, type = c("full", "pos", "neg", "fpt", "separate", "sig"))
saveModelBinSimple(arfmodel)
Arguments

arfmodel  An object of class model.

type  What to save: 'full' = Full model (1 file), 'pos+neg' = positive + negative regions (2 files), 'pos' = Only positive regions (1 file), 'neg' = Only negative regions (1 file), 'fpn' = Full, positive and negative regions (3 files), 'separate' = Individual files for each region (#R files), 'sig' = Only significant regions (1 file).

Details

When type is not 'full', model predictions are not corrected for overlapping regions.

Value

Returns a model object.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

model, writeData, fmri.data

Description

Save a registration object

Usage

saveReg(registration)

Arguments

registration  Object of class registration.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

registration, loadReg
Description

The arf3DS4 "sequence" class contains information of a sequence of fitted models. It can be used to assess the relative fit between models and decide which model fits best.

Objects

Objects can be created by calls of the form `new("sequence", ...)`.

Slots

- `best`: Which model currently fits the best?
- `current`: The current model number in the sequence (not used).
- `regions`: Vector of number-of-regions to in the fitted sequence.
- `mnames`: Vector of model names of the sequence.
- `fit`: Vector of fit indices for each model.
- `minimum`: Vector of minima for each model.
- `valid`: Logical vector indicating valid models.

Methods

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

- `access`: `.classname.slotname(object)`
- `replace`: `.classname.slotname(object)<-`

In this case `classname` is "sequence" and `slotname` is for example "fit". So to access the "fit" slot of an r-object "mysequence" (of class "sequence") type `.sequence.fit(mysequence)`.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

- `minBIC`, `fitRange`

Examples

- `showClass("sequence")`
setFuncTimings

**setFuncTimings**

*Set the timings of the stimuli and link to raw time-series.*

**Description**

Set the timings of the stimuli (i.e. single trials) and link these timings to raw time-series. For each run of each condition the timings of when trials were presented must be given for ARF to determine the single-trial data.

**Usage**

```r
setFuncTimings(subject, condition, run, timings, func_data = NULL, experiment = NULL)
```

**Arguments**

- `subject` Name of the subject.
- `condition` Name of the condition.
- `run` Which run, can be numeric or a filename.
- `timings` A vector with timings (in seconds), possibly with an attribute `stimlen` (vector of stimulus lengths).
- `func_data` Name of the raw time-series datafile (located in the `/subject/funcs` directory).
- `experiment` An optional experiment object.

**Details**

An object of class `functional` is also saved in the appropriate directory.

**Value**

Returns an object of class `functional`.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

- `functional`, `experiment`, `makeSingleTrialEvents`
**setIsoContour**  
*Create isocontours*

**Description**
Create isocontours for each region in the ARF model.

**Usage**
```r
setIsoContour(arfmodel, conf.int = 95)
```

**Arguments**
- **arfmodel**  
  An ARF model object (class `model`).
- **conf.int**  
  Confidence Interval (in percentage) for the isocontour.

**Value**
An object of class `fmri.data` with a volume for each region in the spatial model. Each volume indicates the voxels within the isocontour for that specific region.

**Author(s)**
Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**
`model`, `fmri.data`, `roiConnectivity`

---

**setRegFiles**  
*Set registration files.*

**Description**
Link the files necessary for registration to the appropriate slots in the registration object. Registration requires a working copy of FSL. Files must be copied from the FSL directory to the `/registration` directory of the ARF experiment. By default the standard filenames used by FSL are used.

**Usage**
```r
setRegFiles(registration, examp2stand = "example_func2standard.mat", 
            examp2high = "example_func2highres.mat", high2stand = "highres2standard.mat", 
            example_func = "example_func.nii.gz", highres = "highres.nii.gz", 
            standard = "standard.nii.gz")
```
Arguments

registration Object of class registration to link to.
examp2stand ARF to standard space affine transformation matrix file.
examp2high ARF to high resolution affine transformation matrix file.
high2stand High resolution to standard space affine transformation matrix file.
example_func Nifti file in ARF native space.
highres T1 weighted structural image.
standard Standard space image (MNI152).

Value

Returns a registration object with the appropriate slots filled.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

References

See http://www.fmrib.ox.ac.uk/fsl/ on how to obtain FSL and licences.

See Also

registration, createRegs, setRegParams

**setRegParams**

*Set the registration parameters.*

Description

Set the registration matrices to the appropriate values given the registration files. Registration requires a working copy of FSL.

Usage

setRegParams(registration)

Arguments

registration Object of class registration with appropriate filenames.

Details

First run setRegFiles to set the correct filenames.
**settings-class**

**Value**

Returns a registration object with the appropriate slots filled.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**References**

See [http://www.fmrib.ox.ac.uk/fsl/](http://www.fmrib.ox.ac.uk/fsl/) on how to obtain FSL and licences.

**See Also**

registration, setRegFiles, createRegs

---

<table>
<thead>
<tr>
<th>settings-class</th>
<th>arf3DS4 &quot;settings&quot; class</th>
</tr>
</thead>
</table>

**Description**

The arf3DS4 class "settings" contains the names of all directories and files used by the arf3DS4 package. Default values are available for the "settings" class and adjusting the slot values is usually not necessary. An object of class "settings" is contained in the experiment class.

**Objects**

Objects can be created by calls of the form new("settings", ...).

**Slots**

- **expRda**: Name of the .Rda file containing the experiment information.
- **optionsRda**: Name of the .Rda file containing the options for fitting an ARF model.
- **startRda**: Name of the .Rda file containing the starting values for fitting an ARF model.
- **dataRda**: Name of the .Rda file containing the fMRI data location and information.
- **modelRda**: Name of the .Rda file containing the ARF model information.
- **statsRda**: Name of the .Rda file containing the statistics information of a model (not yet implemented).
- **regRda**: Name of the .Rda file containing the registration information of fMRI data image.
- **funcRda**: Name of the .Rda file containing the functional information of an fMRI data image.
- **subjectPrefix**: Prefix for the subject names. Defaults to empty.
- **conditionPrefix**: Prefix for the condition names. Defaults to empty.
- **modelPrefix**: Prefix for the model names. Defaults to empty.
- **subjectDir**: Name of the subjects directory.
- **conditionDir**: Name of the conditions directory.
dataDir: Name of the data directory.
weightsDir: Name of the weights directory (within conditions/data).
avgDir: Name of the average data directory (within conditions/data).
regDir: Name of the registration directory (within conditions/data).
funcDir: Name of the functional directories (within subjects and conditions)/data.
betaDir: Name of the beta directory (within conditions/data).
modelDir: Name of the model directory.
statsDir: Name of the stats directory (not yet implemented).
modeldatDir: Name of the data directory (within models).
avgdatFile: Name of the averaged data file (fMRI).
avgWFile: Name of the averaged weight file (fMRI).
avgtstatFile: Name of the averaged t-values file (fMRI).
modelDatafile: Name of the model prediction file (fMRI).
modelnamesRda: Name of the modelnames.Rda file containing the fitted models.
residualFile: Name of the residuals file (binary).
derivativeFile: Name of the derivatives file (binary).
weightFile: Name of the weight file (binary).
lowresFile: Name of the lowres structural image.
lowresAvg: Name of the average lowres structural image.
logFile: Name of the logFile (not yet implemented).
version: Object of class "version" (see version)

Methods

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

access .classname.slotname(object)
replace .classname.slotname(object)<-

In this case classname is "settings" and slotname is for example "avgdatfile". So to access the "avgdatfile" slot of an r-object "mysettings" (of class "settings") type .settings.avgdatfile(mysettings).

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

experiment

Examples

showClass("settings")
showModels

List all models for a subject/condition.

Description

List all models for a subject/condition.

Usage

showModels(subject, condition, experiment = NULL)

Arguments

subject Name of the subject.
condition Name of the condition.
experiment Optional object of class "experiment".

Value

Returns an object of class mnames with modelinformation.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

mnames, experiment, loadModel

varcov

Calculate (co)variance matrix of an ARF model.

Description

Calculate (co)variance matrix of the parameters in an ARF model using Sandwich estimation. The method of Sandwich estimation can be modified via the options object of the ARF model.

Usage

varcov(arfmodel)

Arguments

arfmodel A valid ARF model object.
Details

`varcov` needs first-order derivatives and residual matrices, if they do not exist, they are created automatically.

Value

Returns an object of class "model" with the appropriate slots.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

`model`, `processModel`, `makeDerivs`, `options`

---

### version-class

**arf3DS4 "version" class**

**Description**

The "version" class contains information on the version of the arf3DS4 objects. Each arf3DS4 object contains this class.

**Objects**

Objects can be created by calls of the form `new("version", ...)`.  

**Slots**

- `version`: Version number of the arf3DS4 package.
- `build`: Build number of the arf3DS4 package.
- `update`: Update number of the arf3DS4 package.
- `svnrev`: SVN revision on r-forge.

**Methods**

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

`access` : `classname.slotname(object)`

`replace` : `classname.slotname(object)<-`

In this case `classname` is "version" and `slotname` is for example "svnrev". So to access the "svnrev" slot of an r-object "currentversion" (of class "version") type `.version.svnrev(currentversion)`.

**Note**

The "version" class is mainly used for back-compatibility issues.
Author(s)
Wouter D. Weeda - w.d.weeda@gmail.com

Examples

showClass("version")

wald

*Calculate Wald statistics for an arf model.*

Description

Calculate Wald statistics for an arf model.

Usage

wald(arfmodel, waldobject = new("wald"), options = loadOptions(arfmodel))

Arguments

arfmodel A valid model object.
waldobject An object of class "wald". By default a new (empty) "wald" object is created. To test specific hypotheses a "wald" object with the @consts modified can be used.
options An object of class "options".

Details

wald requires the (co)variance matrix of parameter estimates, it will give an error when the arfmodel@varcov slot does not exist (or has invalid values).

Value

Returns an object of class "model" with the appropriate slots.

Author(s)

Wouter D. Weeda - w.d.weeda@gmail.com

See Also

model, varcov, processModel, options
**Description**

The arf3DS4 "wald" class contains information of the statistics on the spatial model parameters. It is used in the `model` class.

**Objects**

Objects can be created by calls of the form `new("wald", ...)`. 

**Slots**

- `consts`: The #regions by #hypothesis matrix defining the constants used for testing.
- `stats`: The #regions by #hypothesis matrix of wald statistics.
- `df1`: Vector of the Degrees of Freedom of the model.
- `df2`: Vector of the Degrees of Freedom of the error.
- `pvalues`: The #regions by #hypothesis matrix of p-values.

**Methods**

Standard arf3DS4 accessor and replacement functions can be used to access and replace slots.

```r
access .classname.slotname(object)
replace .classname.slotname(object)<-
```

In this case `classname` is "wald" and `slotname` is for example "stats". So to access the "stats" slot of an r-object "mywald" (of class "wald") type `.wald.stats(mywald)`.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**See Also**

`model`, `wald`

**Examples**

```r
showClass("wald")
```
writeData  

**Write NIFTI/ANALYZE files.**

**Description**

Write a data-vector to a NIFTI/ANALYZE file.

**Usage**

```
writeData(headinf, datavec)
```

**Arguments**

- **headinf**: An object of class "fmri.data".
- **datavec**: An optional data-vector to be written to the nifti/analyze file. If no datavec is given the @datavector slot of the fmri.data-object is used.

**Details**

All information on filetype @filetype, path @fullpath, and filename @filename are within the fmri.data-object (also see nifti.fileinfo). Changing these values will modify where and how the data are saved.

**Value**

Returns TRUE when successful.

**Author(s)**

Wouter D. Weeda - w.d.weeda@gmail.com

**References**


**See Also**

- fmri.data
- nifti.header
- nifti.fileinfo
- readData
Index

*Topic classes
arfcorrelation-class, 5
data-class, 11
eperiment-class, 14
fmri.data-class, 20
functional-class, 23
mnames-class, 37
model-class, 38
nifti.fileinfo-class, 41
nifti.header-class, 42
options-class, 45
registration-class, 50
sequence-class, 56
settings-class, 60
version-class, 63
wald-class, 65

*Topic datasets
arf.example-data, 4

*Topic file
loadRda, 30

*Topic methods
createRegs, 10
readData, 49
saveModel, 54
saveModelBin, 54
setFuncTimings, 57
setRegFiles, 58
setRegParams, 59
showModels, 62
writeData, 66

*Topic models
arfversion, 6
BIC, 6
checkSolution, 7
clearWarnings, 8
createAllAverages, 8
createAverages, 9
determineStartRect, 12
differenceCor, 13
euclidDist, 14
extr-repl-fmridata, 16
fitConnectivity, 17
fitModel, 18
fitRange, 19
fmri2array, 22
getAtlasLabels, 24
getExp, 25
getModelAtlas, 25
loadData, 26
loadExp, 27
loadFunc, 28
loadModel, 28
loadOptions, 29
loadReg, 30
loadStart, 31
makeDerivs, 32
makeExpDirs, 32
makeLowResStruct, 33
makeROImask, 34
makeSingleTrialEvents, 35
mcpCorrect, 36
minBIC, 37
newModel, 40
processModel, 46
processSeed, 47
pruneModel, 48
readData, 49
readDerivs, 50
RMSEA, 52
roiConnectivity, 52
saveFunc, 53
saveModel, 54
saveModelBin, 54
saveReg, 55
setFuncTimings, 57
setIsoContour, 58
setRegFiles, 58
setRegParams, 59
showModels, 62  
varcov, 62  
wald, 64  
writeData, 66

*Topic package  
arf3DS4-package, 3  
.arfcorrelation.corr, arfcorrelation-method  
(arfcorrelation-class), 5  
.data.dataHeader<-, data-method  
(data-class), 11  
.data.fullpath<-, data-method  
(data-class), 11  
.data.fullpath, data-method  
(data-class), 11  
.data.funcDir, data-method (data-class), 11  
.data.funcDir<-, data-method (data-class), 11  
.data.fullpath, data-method (data-class), 11  
.data.funcRda, data-method (data-class), 11  
.data.n<-, data-method (data-class), 11  
.data.name, data-method (data-class), 11  
.data.name<-, data-method (data-class), 11  
.data.n, data-method (data-class), 11  
.data.p<-, data-method (data-class), 11  
.data.regDir, data-method (data-class), 11  
.data.regDir<-, data-method (data-class), 11  
.data.regDir, data-method (data-class), 11  
.data.regDir<-, data-method (data-class), 11  
.data.version, data-method (data-class), 11  
.data.version<-, data-method (data-class), 11  
.data.weightfiles, data-method (data-class), 11  
.data.weightfiles<-, data-method (data-class), 11  
.example (arf-example-data), 4  
.experiment.avgDir, experiment-method (experiment-class), 14  
.experiment.avgDir<-, experiment-method (experiment-class), 14
.experiment.optionsRda, experiment-method (experiment-class), 14
.experiment.optionsRda <-, experiment-method (experiment-class), 14
.experiment.path, experiment-method (experiment-class), 14
.experiment.path <-, experiment-method (experiment-class), 14
.experiment.regDir, experiment-method (experiment-class), 14
.experiment.regDir <-, experiment-method (experiment-class), 14
.experiment.residualFile, experiment-method (experiment-class), 14
.experiment.residualFile <-, experiment-method (experiment-class), 14
.experiment.startRda, experiment-method (experiment-class), 14
.experiment.startRda <-, experiment-method (experiment-class), 14
.experiment.statsDir, experiment-method (experiment-class), 14
.experiment.statsDir <-, experiment-method (experiment-class), 14
.experiment.statsRda, experiment-method (experiment-class), 14
.experiment.statsRda <-, experiment-method (experiment-class), 14
.experiment.subject.names, experiment-method (experiment-class), 14
.experiment.subject.names <-, experiment-method (experiment-class), 14
.experiment.subject.num, experiment-method (experiment-class), 14
.experiment.subject.num <-, experiment-method (experiment-class), 14
.experiment.subjectDir, experiment-method (experiment-class), 14
.experiment.subjectDir <-, experiment-method (experiment-class), 14
.experiment.subjectPrefix, experiment-method (experiment-class), 14
.experiment.subjectPrefix <-, experiment-method (experiment-class), 14
.experiment.version, experiment-method (experiment-class), 14
.experiment.version <-, experiment-method (experiment-class), 14
.experiment.weightFile, experiment-method (experiment-class), 14
.experiment.weightFile <-, experiment-method (experiment-class), 14
.fmri.data.aux_file, fmri.data-method (fmri.data-class), 20
.fmri.data.aux_file <-, fmri.data-method (fmri.data-class), 20
.fmri.data.bitpix, fmri.data-method (fmri.data-class), 20
.fmri.data.bitpix <-, fmri.data-method (fmri.data-class), 20
.fmri.data.cal_max, fmri.data-method (fmri.data-class), 20
.fmri.data.cal_max <-, fmri.data-method (fmri.data-class), 20
.fmri.data.cal_min, fmri.data-method (fmri.data-class), 20
.fmri.data.cal_min <-, fmri.data-method (fmri.data-class), 20
.fmri.data.data.signed, fmri.data-method (fmri.data-class), 20
.fmri.data.data.signed <-, fmri.data-method (fmri.data-class), 20
.fmri.data.data.type, fmri.data-method (fmri.data-class), 20
.fmri.data.data.type <-, fmri.data-method (fmri.data-class), 20
.fmri.data.data_type, fmri.data-method (fmri.data-class), 20
.fmri.data.data_type <-, fmri.data-method (fmri.data-class), 20
.fmri.data.datatype, fmri.data-method (fmri.data-class), 20
.fmri.data.datatype <-, fmri.data-method (fmri.data-class), 20
.fmri.data.datavec, fmri.data-method (fmri.data-class), 20
.fmri.data.datavec <-, fmri.data-method (fmri.data-class), 20
<table>
<thead>
<tr>
<th>fmri.data.db_name, fmri.data-method</th>
</tr>
</thead>
<tbody>
<tr>
<td>fmri.data.db_name=fMRI.data-method</td>
</tr>
<tr>
<td>fmri.data.db_name&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.db_name&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.db_name&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.db_name&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.db_name&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.db_name!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.dim_info, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.dim_info&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.dim_info&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.dim_info&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.dim_info&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.dim_info&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.dim_info!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extension, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extension&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extension&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extension&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extension&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extension&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extension!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.filename, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.filename&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.filename&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.filename&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.filename&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.filename&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.filename!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.extrabytes!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.fullpath, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.fullpath&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.fullpath&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.fullpath&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.fullpath&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.fullpath&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.fullpath!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmax, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmax&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmax&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmax&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmax&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmax&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmax!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmin, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmin&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmin&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmin&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmin&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmin&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.glmin!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzipped, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzipped&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzipped&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzipped&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzipped&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzipped&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzipped!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzip&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzip&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.gzip!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_code, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_code&lt;&gt;=!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_code&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_code&gt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_code&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_code&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_code!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_name, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_name&lt;&gt;=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_name&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_name=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_name&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_name&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_name!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p1, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p1&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p1&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p1=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p1&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p1&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p1!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p2, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p2&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p2&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p2=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p2&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p2&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p2!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p3, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p3&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p3&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p3=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p3&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p3&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.intent_p3!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.magic, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.magic&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.magic&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.magic=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.magic&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.magic&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.magic!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.pixdim, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.pixdim&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.pixdim&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.pixdim=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.pixdim&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.pixdim&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.pixdim!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qform_code, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qform_code&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qform_code&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qform_code=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qform_code&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qform_code&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qform_code!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_x!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_y!=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;&gt;(=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;=, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z=!, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&lt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z&gt;, fmri.data-method</td>
</tr>
<tr>
<td>fmri.data.qoffset_z!=, fmri.data-method</td>
</tr>
</tbody>
</table>
INDEX

.functional.linkedfiles, functional-method
  (functional-class), 23
.functional.linkedfiles<-, functional-method
  (functional-class), 23
.functional.timings, functional-method
  (functional-class), 23
.functional.timings<-, functional-method
  (functional-class), 23
.functional.version, functional-method
  (functional-class), 23
.functional.version<-, functional-method
  (functional-class), 23
.mnames.condition, mnames-method
  (mnames-class), 37
.mnames.condition<-, mnames-method
  (mnames-class), 37
.mnames.experiment, mnames-method
  (mnames-class), 37
.mnames.experiment<-, mnames-method
  (mnames-class), 37
.mnames.mnames, mnames-method
  (mnames-class), 37
.mnames.mnames<-, mnames-method
  (mnames-class), 37
.mnames.subject, mnames-method
  (mnames-class), 37
.mnames.subject<-, mnames-method
  (mnames-class), 37
.model.dataHeader, model-method
  (model-class), 38
.model.dataHeader<-, model-method
  (model-class), 38
.model.derivativeFile, model-method
  (model-class), 38
.model.derivativeFile<-, model-method
  (model-class), 38
.model.estimates, model-method
  (model-class), 38
.model.estimates<-, model-method
  (model-class), 38
.model.fit, model-method
  (model-class), 38
.model.fit<-, model-method
  (model-class), 38
.model.fullmodelDataFile, model-method
  (model-class), 38
.model.fullmodelDataFile<-, model-method
  (model-class), 38
.model.fullpath, model-method
  (model-class), 38
.model.fullpath<-, model-method
  (model-class), 38
.model.funcDir, model-method
  (model-class), 38
.model.funcDir<-, model-method
  (model-class), 38
.model.funcRda, model-method
  (model-class), 38
.model.funcRda<-, model-method
  (model-class), 38
.model.gradient, model-method
  (model-class), 38
.model.gradient<-, model-method
  (model-class), 38
.model.hessian, model-method
  (model-class), 38
.model.hessian<-, model-method
  (model-class), 38
.model.iternates, model-method
  (model-class), 38
.model.iternates<-, model-method
  (model-class), 38
.model.logFile, model-method
  (model-class), 38
.model.logFile<-, model-method
  (model-class), 38
.model.version<-, .model-method (model-class), 38
.nifti.header.aux_file<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.bitpix<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.bitpix, .nifti.header-method (nifti.header-class), 42
.nifti.header.cal_max<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.cal_min<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.data_signed, .nifti.header-method (nifti.header-class), 42
.nifti.header.data.type, .nifti.header-method (nifti.header-class), 42
.nifti.header.data.type<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.db_name<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.db_name, .nifti.header-method (nifti.header-class), 42
.nifti.header.data.type<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.data.type, .nifti.header-method (nifti.header-class), 42
.nifti.header.data_type<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.data_type, .nifti.header-method (nifti.header-class), 42
.nifti.header.datatype<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.datatype, .nifti.header-method (nifti.header-class), 42
.nifti.header.descrip<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.descrip, .nifti.header-method (nifti.header-class), 42
.nifti.header.fullpath<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.fullpath, .nifti.header-method (nifti.header-class), 42
.nifti.header.dims<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.dims, .nifti.header-method (nifti.header-class), 42
.nifti.header.version<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.version, .nifti.header-method (nifti.header-class), 42
.nifti.header.aux_file<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.endian<-, .nifti.header-method (nifti.header-class), 42
.nifti.header.endian, .nifti.header-method (nifti.header-class), 42
INDEX

.. nifti.header.scl_inter<-, nifti.header.method.nifti.header.toffset<-, nifti.header.method
   (nifti.header-class), 42
   (nifti.header-class), 42
.. nifti.header.scl_slope, nifti.header.method nifti.header.version, nifti.header.method
   (nifti.header-class), 42
   (nifti.header-class), 42
   (nifti.header-class), 42
.. nifti.header.scl_slope<-, nifti.header.method nifti.header.version<-, nifti.header.method
   (nifti.header-class), 42
   (nifti.header-class), 42
   (nifti.header-class), 42
.. nifti.header.session_error, nifti.header.method nifti.header.vox_offset, nifti.header.method
   (nifti.header-class), 42
   (nifti.header-class), 42
   (nifti.header-class), 42
.. nifti.header.session_error<-, nifti.header.method nifti.header.vox_offset<-, nifti.header.method
   (nifti.header-class), 42
   (nifti.header-class), 42
   (nifti.header-class), 42
.. nifti.header.sform_code, nifti.header.method nifti.header.xyz_units, nifti.header.method
   (nifti.header-class), 42
   (nifti.header-class), 42
   (nifti.header-class), 42
.. nifti.header.sform_code<-, nifti.header.method nifti.header.xyz_units<-, nifti.header.method
   (nifti.header-class), 42
   (nifti.header-class), 42
   (nifti.header-class), 42
.. nifti.header.sizeof_hdr, nifti.header.method .options.adjust.n, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.sizeof_hdr<-, nifti.header.method options.adjust.n<-, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.slice_code, nifti.header.method .options.chk.method, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.slice_code<-, nifti.header.method options.chk.method<-, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.slice_duration, nifti.header.method options.chk.range, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.slice_duration<-, nifti.header.method options.chk.range<-, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.slice_end, nifti.header.method .options.min.analyticalgrad, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.slice_end<-, nifti.header.method .options.min.analyticalgrad<-, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.slice_start, nifti.header.method .options.min.boundlim, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.slice_start<-, nifti.header.method .options.min.boundlim<-, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.srow_x, nifti.header.method .options.min.iterlim, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.srow_x<-, nifti.header.method .options.min.iterlim<-, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.srow_y, nifti.header.method .options.min.routine, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.srow_y<-, nifti.header.method .options.min.routine<-, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.srow_z, nifti.header.method .options.nlm.gradtol, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.srow_z<-, nifti.header.method .options.nlm.gradtol<-, options-method
   (nifti.header-class), 42
   (options-class), 45
.. nifti.header.toffset, nifti.header.method .options.nlm.steptol, options-method
   (nifti.header-class), 42
   (options-class), 45
INDEX
<table>
<thead>
<tr>
<th>Field</th>
<th>Method</th>
<th>Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>settings_lowresAvg</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_lowresFile</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelDataFile</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelDataFile</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelPrefix</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelPrefix</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelDatDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelDatDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelNamesRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_modelNamesRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_optionsRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_optionsRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_regDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_regDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_regDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_regDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_regRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_regRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_residualFile</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_residualFile</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_startRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>settings_startRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>statsDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>statsDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>statsDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>statsRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>statsRda</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>subjectDir</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>subjectPrefix</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>subjectPrefix</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>version</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>version</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>version</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>version</td>
<td>method</td>
<td>class</td>
<td>60</td>
</tr>
<tr>
<td>wald_consts</td>
<td>method</td>
<td>class</td>
<td>65</td>
</tr>
</tbody>
</table>
.waldconsts<-, wald-method
  (wald-class), 65
.wald.df1<-, wald-method (wald-class), 65
.wald.df1<-, wald-method (wald-class), 65
.wald.df2<-, wald-method (wald-class), 65
.wald.df2<-, wald-method (wald-class), 65
.wald.pvalues<-, wald-method (wald-class), 65
.wald.pvalues<-, wald-method (wald-class), 65

[, fmri.data-method
  (extr-repl-fmridata), 16
[<-, fmri.data-method
  (extr-repl-fmridata), 16

arf-example-data, 4
arf3DS4 (arf3DS4-package), 3
arf3DS4-package, 3
arfcorrelation, 17, 53
arfcorrelation-class, 5
arfversion, 6

BIC, 6, 52
checkSolution, 7
clearWarnings, 8
creatAllAverages, 8, 10
creatAverages, 8, 9, 9, 11
creatReg, 10, 59, 60
data, 4, 9, 10, 26, 33, 34, 40
data-class, 11
determineStartRect, 12
determineStartRectSimple
  (determineStartRect), 12
differenceCor, 13
euclidDist, 14
experiment-class, 14
eextr-repl-fmridata, 16

fitConnectivity, 4, 5, 17, 53
fitModel, 4, 18, 46–48
fitModelOptim (fitModel), 18
fitRange, 19, 56

fitSimpleModelOptim (fitModel), 18
fmri.data, 3, 16–19, 22, 34–37, 42, 44, 46, 47, 49, 53, 55, 58, 66
fmri.data-class, 20
fmri2array, 17, 22
functional, 28, 35, 53, 57
functional-class, 23

getAtlasLabels, 24, 25, 26
getExp, 4, 14, 16, 25
getModelAtlas, 25

loadData, 26
loadExp, 4, 9, 10, 14, 16, 25, 26, 27, 33
loadFunc, 28, 53
loadModel, 28, 62
loadOptions, 4, 29
loadRDa, 30
loadReg, 30, 55
loadStart, 31

makeDerivs, 32, 63
makeExample, 4
makeExpDir, 3, 14, 16, 32
makeLowResStruct, 33
makeLowResStructAvg (makeLowResStruct), 33
makeROImask, 34
makeSingleTrialEvents, 4, 17, 28, 35, 57
mcpCorrect, 36
minBIC, 37, 56
mnames, 29, 62
mnames-class, 37
model-class, 38

newModel, 4, 18, 19, 40, 47, 48
nifti.fileinfo, 21, 22, 44, 66
nifti.fileinfo-class, 41
nifti.header, 20–22, 42, 66
nifti.header-class, 42

optim, 18
options, 4, 7, 12, 13, 18, 19, 29, 30, 39–41, 46–49, 52, 62–64
options-class, 45
INDEX

plot, fmri.data, missing-method
  (fmri.data-class), 20
plot, model, missing-method
  (model-class), 38
processModel, 46, 63, 64
processSeed, 47, 49
pruneModel, 48, 48

readData, 49, 66
readDerivs, 50
registration, 10, 24, 31, 55, 59, 60
registration-class, 50
RMSEA, 7, 52
roiConnectivity, 5, 17, 52, 58

saveFunc, 53
saveModel, 54
saveModelBin, 54
saveModelBinSimple (saveModelBin), 54
saveOptions, 4
saveOptions (loadOptions), 29
saveReg, 55
saveStart (loadStart), 31
sequence, 19, 37
sequence-class, 56
setFuncTimings, 23, 28, 35, 57
setIsoContour, 53, 58
setRegFiles, 10, 24, 33, 34, 58, 59, 60
setRegParams, 10, 33, 34, 59, 59
settings, 14, 16, 33
settings-class, 60
show, data-method (data-class), 11
show, experiment-method
  (experiment-class), 14
show, fmri.data-method
  (fmri.data-class), 20
show, mnames-method (mnames-class), 37
show, model-method (model-class), 38
show, options-method (options-class), 45
show, sequence-method (sequence-class), 56
show, version-method (version-class), 63
showModels, 62
summary, fmri.data-method
  (fmri.data-class), 20

varcov, 4, 46, 47, 62, 64
version, 6, 11, 16, 21, 23, 39, 42, 44, 45, 51, 61
version-class, 63
wald, 4, 39, 40, 46, 47, 64, 65
wald-class, 65
writeData, 55, 66