

Package ‘tractor.base’

March 13, 2012

Version 2.1.0

Date 2012-03-09

Title A package for reading, manipulating and visualising magnetic resonance images

Author Jon Clayden <jon.clayden+tractor@gmail.com>

Maintainer Jon Clayden <jon.clayden+tractor@gmail.com>

Depends R (>= 2.12.1), graphics, grDevices, methods, stats, utils

Imports reportr

Suggests oro.nifti

Description The tractor.base package consists of functions for working with magnetic resonance images. It can read and write image files stored in Analyze, NIfTI, MGH and DICOM formats (DICOM support is read only), generate images for use as regions of interest, and manipulate and visualise images.

Encoding UTF-8

LazyLoad yes

License GPL-2

URL <https://github.com/jonclayden/tractor/>

Repository CRAN

Date/Publication 2012-03-13 16:58:50

R topics documented:

data types	2
DicomMetadata-class	3
dictionary	5
equivalent	5
execute	6
implode	7
MriImage-class	8
MriImageMetadata-class	10
neighbourhoodInfo	12
newDicomMetadataFromFile	13
newMriImageFromDicom	14
newMriImageFromFile	15
newMriImageMetadataFromTemplate	17
newMriImageWithData	18
newMriImageWithDataRepresentation	20
newSparseArrayWithData	21
nilObject	21
path manipulation	22
printLabelledValues	23
promote	24
SerializableObject-class	25
serialisation	26
sortDicomDirectory	27
SparseArray-class	28
SparseOrDenseArray-class	29
threadSafeTempFile	30
vector functions	31
visualisation	32
Index	34

data types

MriImage data types

Description

Convert a numeric type code into the image data type specifier used in `MriImage` objects.

Usage

```
getDataByNiftiCode(code)
```

Arguments

code A numeric value indicating the data type code required. Supported values in TractoR are 2 (8-bit unsigned integer), 4 (16-bit signed integer), 8 (32-bit signed integer), 16 (32-bit floating point), 64 (64-bit floating point), 256 (8-bit signed integer), 512 (16-bit unsigned integer) and 768 (32-bit unsigned integer).

Value

A list with the following elements

type An R storage mode, either integer or double.
size The number of bytes (not bits) per voxel in this data type.
isSigned Logical value indicating whether data values are signed.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[MriImage](#), and the NIfTI-1 standard (<http://nifti.nimh.nih.gov/nifti-1>).

DicomMetadata-class *Class "DicomMetadata"*

Description

This class represents DICOM metadata, which typically contains detailed information about the scan parameters and subject.

Extends

Class "[SerializableObject](#)", directly.

All reference classes extend and inherit methods from "[envRefClass](#)".

Fields

source: Object of class character.
tags: Object of class data.frame.
tagOffset: Object of class integer.
dataOffset: Object of class integer.
dataLength: Object of class integer.
explicitTypes: Object of class logical.
endian: Object of class character.

Class-Based Methods

getTags(): Retrieve the data frame containing all available tags, which has columns "groups", "elements", "types" and "values". See the DICOM standard for more information.
getTagValue(group, element): Retrieve the value of the tag with specified group and element numbers, usually given in hex.
getTagOffset(): Retrieve the byte offset of useful tags in the file.
getDataOffset(): Retrieve the byte offset of the data in the DICOM file.
getDataLength(): Retrieve the length of the data part of the DICOM file in bytes.
getSource(): Retrieve the name of the source file.
getEndianness(): Retrieve the endianness of the file: "big" or "little".
nTags(): Retrieve the number of tags stored in this object.

The following methods are inherited (from the corresponding class): serialise ("SerialisableObject"), methods ("SerialisableObject")

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

Examples

```
showClass("DicomMetadata")
```

dictionary	<i>DICOM tag dictionary</i>
------------	-----------------------------

Description

This data set provides information about DICOM tags, including descriptions and data types. It is primarily for internal use.

Usage

```
dictionary
```

Format

A data frame, with rows corresponding to known tags.

equivalent	<i>Test two numeric vectors for equivalence</i>
------------	---

Description

Test two numeric vectors for equivalence.

Usage

```
equivalent(x, y, signMatters = TRUE, ...)
```

Arguments

x	The first numeric vector.
y	The second numeric vector.
signMatters	Logical value: if FALSE then equivalence in absolute value is sufficient.
...	Additional arguments to all.equal , notably tolerance.

Details

This function is a wrapper for `isTRUE(all.equal(x,y,...))`, but with the additional capability of doing sign-insensitive comparison.

Value

TRUE if all elements of x match all elements of y to within tolerance, ignoring signs if required. FALSE otherwise.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[all.equal](#)

Examples

```
equivalent(c(-1,1), c(1,1)) # FALSE
equivalent(c(-1,1), c(1,1), signMatters=FALSE) # TRUE
equivalent(1:2, 2:3, tolerance=2) # TRUE
```

execute

Find or run an external executable file

Description

The execute function is a wrapper around the [system](#) function in base R, which echoes the command being run (including the full path to the executable) if the reportr output level is Debug. `locateExecutable` simply returns the path to an executable file on the system PATH.

Usage

```
execute(executable, paramString = NULL, errorOnFail = TRUE, silent = FALSE, ...)
locateExecutable(fileName, errorIfMissing = TRUE)
```

Arguments

<code>executable</code> , <code>fileName</code>	Name of the executable to run.
<code>paramString</code>	A character string giving the parameters to pass to the executable, if any.
<code>errorOnFail</code> , <code>errorIfMissing</code>	Logical value: should an error be produced if the executable can't be found?
<code>silent</code>	Logical value: should the executable be run without any output?
<code>...</code>	Additional arguments to system .

Value

For `execute`, the return value of the underlying call to `system`. For `locateExecutable`, the location of the requested executable, or `NULL` if it could not be found.

Note

These functions are designed for Unix systems and may not work on Windows.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[system](#)

implode	<i>Create a character string by concatenating the elements of a vector</i>
---------	--

Description

Create a character string by concatenating the elements of a vector, using a separator and optional final separator.

Usage

```
implode(strings, sep = "", finalSep = NULL)
```

Arguments

<code>strings</code>	A vector, which will be coerced to mode character.
<code>sep</code>	A unit length character vector giving the separator to insert between elements.
<code>finalSep</code>	An optional unit length character vector giving the separator to insert between the final two elements.

Value

A character vector of length one.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[paste](#)

Examples

```
implode(1:3, ", ") # "1, 2, 3"
implode(1:3, ", ", " and ") # "1, 2 and 3"
implode(1:2, ", ", " and ") # "1 and 2"
```

MriImage-class	Class "MriImage"
----------------	------------------

Description

This class represents an MRI image. An object of this class is made up of some voxel data, stored as a sparse or dense numeric array, and an [MriImageMetadata](#) object, which stores extra information about the image, such as the file it was read from, the voxel dimensions, and so on. Since the class inherits from [MriImageMetadata](#), any object can be treated as an object of that class where needed. The group generic functions [Math](#), [Ops](#) and [Summary](#) are defined for this class, as are methods for coercing to and from a standard R [array](#).

Extends

Class "[MriImageMetadata](#)", directly. Class "[SerialisableObject](#)", by class "[MriImageMetadata](#)", distance 2.

All reference classes extend and inherit methods from "[envRefClass](#)".

Methods

```
[ signature(x = "MriImage"): ...
[<- signature(x = "MriImage"): ...
```

Fields

imagedims: See the `MriImageMetadata` class.
voxdims: See the `MriImageMetadata` class.
voxunit: See the `MriImageMetadata` class.
source: See the `MriImageMetadata` class.
datatype: See the `MriImageMetadata` class.
origin: See the `MriImageMetadata` class.
storedXform: See the `MriImageMetadata` class.
tags: See the `MriImageMetadata` class.
data: The image data, stored as an object of class `SparseOrDenseArray`.

Class-Based Methods

initialize(data, metadata, ...): Create a new object of this class.
summarise(): Retrieve information about this object. This method is usually only called implicitly by the "show" method.
getData(): Retrieve the array of voxel values.
getDataAtPoint(...): Retrieve the value of the voxel at the location specified by `c(...)`. Returns NA if the location is out of bounds.
getMetadata(): Retrieve the embedded `MriImageMetadata` object.
getSparseness(): Retrieve the proportion of image pixels or voxels which are nonzero.
isSparse(): TRUE if the image data is stored as a `SparseArray` object; FALSE otherwise.

The following methods are inherited (from the corresponding class): `getFieldOfView` ("MriImageMetadata"), `getVoxelUnit` ("MriImageMetadata"), `getDimensions` ("MriImageMetadata"), `initialize` ("MriImageMetadata"), `getDimensionality` ("MriImageMetadata"), `isInternal` ("MriImageMetadata"), `getDataType` ("MriImageMetadata"), `getSource` ("MriImageMetadata"), `getTags` ("MriImageMetadata"), `serialise` ("SerializableObject"), `getVoxelDimensions` ("MriImageMetadata"), `getStoredXformMatrix` ("MriImageMetadata"), `summarise` ("MriImageMetadata"), `getTag` ("MriImageMetadata"), `setSource` ("MriImageMetadata"), `getOrigin` ("MriImageMetadata"), `methods` ("SerializableObject")

Note

The Summary group generic currently works only for a single image argument. A call such as `max(image1, image2)` will produce an error.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

The [SerialisableObject](#) and [MriImageMetadata](#) classes, which this class extends. Also the group generic functions [Math](#), [Ops](#) and [Summary](#) (in the methods package); [newMriImageWithSimpleFunction](#) and [newMriImageWithBinaryFunction](#).

Examples

```
showClass("MriImage")
```

```
MriImageMetadata-class
      Class "MriImageMetadata"
```

Description

This class represents MRI image metadata. An object of this class contains various information about an image, such as its image and voxel dimensions, coordinate origin and storage data type. An object of type [MriImage](#) can be treated as a metadata object, since the latter is embedded in it.

Extends

Class "[SerialisableObject](#)", directly.

All reference classes extend and inherit methods from "[envRefClass](#)".

Fields

`imagedims`: Object of class integer.
`voxdims`: Object of class numeric.
`voxunit`: Object of class character.
`source`: Object of class character.
`datatype`: Object of class list.
`origin`: Object of class numeric.
`storedXform`: Object of class matrix.
`tags`: Object of class list.

Class-Based Methods

`initialize(imagedims, voxdims, voxunit, source, datatype, origin, storedXform, tags, ...)`: Create a new object of this class.
`summarise()`: Retrieve information about this object. This method is usually only called implicitly by the "show" method.
`getFieldOfView()`: Retrieve the field of view along each dimension, i.e. the product of the image dimensions and voxel dimensions.
`getDimensions()`: Retrieve an integer vector giving the dimensions of the image in voxels.

- `getDimensionality()`: Retrieve an integer vector of length 1 giving the number of dimensions in the image.
- `getVoxelDimensions()`: Retrieve a numeric vector giving the dimensions of the image voxels in spatial units, typically mm.
- `getVoxelUnit()`: Retrieve the spatial unit used by `getVoxelDimensions()`. The most common value is "mm". Can be the empty string if undefined.
- `getDataType()`: Retrieve a list describing the storage data type of the image. See [getDataTypeByNiftiCode](#) for details.
- `getOrigin()`: Retrieve the coordinate origin of the image.
- `getSource()`: Retrieve a character vector of length 1 giving the source file for the image. Returns "internal" if the image was not read from a file.
- `getStoredXformMatrix()`: Retrieve the "xform" matrix stored with the image. If the image was created internally, the result will be NA.
- `getTags()`: Retrieve any tags associated with the image, as a list of keys and values.
- `getTag(key)`: Retrieve the value associated with the specified key. If the key is not present, NA will be returned.
- `setSource(newSource)`: Set the source of the image.
- `isInternal()`: TRUE if the image was generated by a function, rather than read from a file.

The following methods are inherited (from the corresponding class): `serialise` ("SerialisableObject"), `methods` ("SerialisableObject")

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

Examples

```
showClass("MriImageMetadata")
```

neighbourhoodInfo *Class representing a cuboidal neighbourhood in an image*

Description

This class represents a cuboidal region of an image, with a centre and a fixed voxel width.

Usage

```
createNeighbourhoodInfo(width, dim = 3, centre = rep(0, dim))
```

Arguments

width	An integer voxel width. Must be odd.
dim	An integer giving the dimensionality of the neighbourhood. Currently must be 3.
centre	A numeric vector giving the centre voxel of the neighbourhood. Must have exactly dim elements.

Value

createNeighbourhoodInfo returns a list with class "neighbourhoodInfo" and elements

width	Copied from the width argument.
dim	Copied from the dim argument.
centre	Copied from the centre argument.
vectors	dim x width^dim matrix whose columns give the locations of each point in the neighbourhood.
innerProducts	A square, symmetric matrix of inner products between every location in the neighbourhood and every other.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

`newDicomMetadataFromFile`*Read a DICOM file into a DicomMetadata object*

Description

This function reads a DICOM file into a `DicomMetadata` object. Only DICOM files from magnetic resonance scanners are supported.

Usage

```
newDicomMetadataFromFile(fileName, checkFormat = TRUE, dictionary = NULL, stopTag = NULL)
```

Arguments

<code>fileName</code>	The name of a DICOM file.
<code>checkFormat</code>	If TRUE, the function will check for the magic string "DICM" at byte offset 128. This string should be present, but in reality not all files contain it.
<code>dictionary</code>	A tag dictionary to use when reading the file. If NULL then the built-in <code>dictionary</code> will be loaded and used.
<code>stopTag</code>	An integer vector giving the group and element numbers (in that order) of a DICOM tag, or NULL. If not NULL, the function will stop parsing the DICOM file if the specified tag is encountered. This can be used to speed up the process if a specific tag is required.

Value

`newDicomMetadataFromFile` returns a `DicomMetadata` object, or NULL on failure.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

The DICOM standard, found online at <http://dicom.nema.org/>. (Warning: may produce headaches!) Also `dictionary`, and `newMriImageFromDicom` for information on how to create `MriImage` objects from DICOM files.

newMriImageFromDicom *Functions for reading images from DICOM files*

Description

Functions for reading images from DICOM files.

Usage

```
newMriImageFromDicom(fileName)
newMriImageFromDicomDirectory(dicomDir, readDiffusionParams = FALSE)
newMriImageFromDicomMetadata(metadata, flipY = TRUE)
newMriImageMetadataFromDicom(fileName)
newMriImageMetadataFromDicomMetadata(dicom)
```

Arguments

fileName	Character vector of length one giving the name of a DICOM file.
dicomDir	Character vector of length one giving the name of a directory containing DICOM files.
readDiffusionParams	Logical value: should diffusion MRI parameters (b-values and gradient directions) be retrieved from the files if possible?
metadata, dicom	DicomMetadata objects.
flipY	Logical value: should the image be flipped in the Y direction? Usually this is appropriate to convert between DICOM's LPS and Tractor's LAS storage conventions.

Value

newMriImageFromDicom and newMriImageFromDicomMetadata return an MriImage object. newMriImageMetadataFromDicom and newMriImageMetadataFromDicomMetadata return an MriImageMetadata object. newMriImageFromDicomDirectory returns a list containing elements

image	An MriImage object.
bValues	Diffusion b-values, if requested. Will be NA if the information could not be found in the files.
bVectors	Diffusion gradient vectors, if requested. Will be NA if the information could not be found in the files.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[DicomMetadata](#), [MriImage](#), [MriImageMetadata](#), [sortDicomDirectory](#).

newMriImageFromFile *Working with MRI images stored in NiftI, Analyze and MGH formats*

Description

Functions for reading, writing, locating, copying and removing MRI images stored in NiftI, Analyze and MGH formats.

Usage

```
newMriImageFromFile(fileName, fileType = NULL, volumes = NULL)
newMriImageMetadataFromFile(fileName, fileType = NULL)
writeMriImageToFile(image, fileName = NULL, fileType = NA, datatype = NULL, overwrite = TRUE)

identifyImageFileNames(fileName, fileType = NULL, errorIfMissing = TRUE)
imageFileExists(fileName, fileType = NULL)
removeImageFilesWithName(fileName)
copyImageFiles(from, to, overwrite = FALSE, deleteOriginals = FALSE)
symlinkImageFiles(from, to, overwrite = FALSE, relative = TRUE)
```

Arguments

fileName, from, to	File names, with or without appropriate extension.
image	An MriImage object.
fileType	A character vector of length one, giving the file type required or expected. If this option is missing, the file type used for writing images will be taken from the tractorFileType option. See Details.
volumes	An optional integer vector specifying a subset of volumes to read (generally to save memory). If given, only the requested volumes in the 4D file will be read.
datatype	A storage data type. See getDataByNiftiCode .
overwrite	Logical value: overwrite an existing image file? For writeMriImageToFile, an error will be raised if there is an existing file and this is set to FALSE.
errorIfMissing	Logical value: raise an error if no suitable files were found?

deleteOriginals

Logical value: if TRUE, copyImageFiles performs a move rather than a copy.

relative

Logical value: if TRUE, the path stored in the symlink will be relative (e.g.

"../some_dir/some_image.nii") rather than absolute (e.g. "/path/to/some_dir/some_image.nii")

Details

NIfTI and Analyze are related formats for storing magnetic resonance images. NIfTI is a more recent extension of Analyze, and contains more specific information about, for example, the orientation of the image. Its use is therefore recommended where possible. MGH format is used by the popular image processing package FreeSurfer. These formats use a number of different file extensions, but the details are abstracted away from the user by these functions.

TractoR does not allow for files with the same basic name using multiple Analyze/NIfTI/MGH formats in a single directory (e.g. "foo.nii" AND "foo.img"), and these functions will produce an error if multiple compatible files exist.

Suitable values for fileType (and the tractorFileType option, which is used as a default) are ANALYZE, NIFTI, NIFTI_PAIR (the two-file NIfTI format), MGH, ANALYZE_GZ, NIFTI_GZ, NIFTI_PAIR_GZ and MGH_GZ. The latter four are gzipped versions of the former four. NIFTI_GZ is recommended unless there is a need for one of the others. This is the default value for the tractorFileType option, but that can be changed using a call to [options](#), or by setting the TRACTOR_FILETYPE environment variable before loading the tractor.base package.

Since multiple files may be involved, copying, moving or symlinking images is not trivial. copyImageFiles and symlinkImageFiles are wrappers around the standard functions [file.copy](#) and [file.symlink](#) which handle this complexity.

Value

newMriImageFromFile and newMriImageMetadataFromFile return the appropriate image or metadata object. imageFileExists returns TRUE if an existing file with the specified name exists (all file extensions are checked), and FALSE otherwise. removeImageFilesWithName returns the result of [unlink](#) applied to all relevant files. writeMriImageToFile and identifyImageFileNames return a list with the following elements, describing the identified or written files:

fileStem The file name without extension.

headerFile The full header file name.

imageFile The full image file name.

format The format of the files ("Nifti", "Analyze" or "Mgh"). Not returned by writeMriImageToFile.

copyImageFiles and symlinkImageFiles are called for their side effects.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

The NIfTI-1 standard (<http://nifti.nimh.nih.gov/nifti-1>); [MriImage](#), [MriImageMetadata](#), [getDataByNiftiCode](#) and [unlink](#).

newMriImageMetadataFromTemplate

Duplicates an MriImageMetadata object, possibly with modification

Description

Duplicates an MriImageMetadata object, possibly with modification.

Usage

```
newMriImageMetadataFromTemplate(metadata, imageDims = NA, voxelDims = NA, voxelUnit = NA,
                                datatype = NA, origin = NA, tags = NA)
```

Arguments

metadata	An existing MriImageMetadata object.
imageDims	Image dimensions.
voxelDims	Voxel dimensions.
voxelUnit	Voxel dimension unit, usually "mm".
datatype	Image data type. See getDataByNiftiCode .
origin	Coordinate origin.
tags	A list containing elements keys and values, specifying tags associated with the image.

Value

An MriImageMetadata object, like the one specified in the first argument, but with the specified fields replaced by new values. Any field with value NA (the default except for source) will be taken from the original metadata object.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[MriImageMetadata](#), [MriImage](#), [getDataByNiftiCode](#).

newMriImageWithData *Functions for creating MriImage objects from data*

Description

Functions for creating MriImage objects from data, including other images.

Usage

```
newMriImageWithData(data, metadata)
newMriImageFromTemplate(image, ...)

newMriImageByExtraction(image, dim, loc)
extractDataFromMriImage(image, dim, loc)
newMriImageByMasking(image, mask)
newMriImageByThresholding(image, level, defaultValue = 0)
newMriImageByTrimming(image, clearance = 4)

newMriImageAsShapeOverlay(type = c("cross", "block"), baseImage, ...)
generateImageDataForShape(type = c("cross", "block"), dim, background = 0,
                           centre = NA, width = NA)

newMriImageWithSimpleFunction(image, fun, ..., newDataType = NULL)
newMriImageWithBinaryFunction(image1, image2, fun, ..., newDataType = NULL)
```

Arguments

data	An array of voxel data.
metadata	An MriImageMetadata object.
image, image1, image2	MriImage objects.
dim, loc	For newMriImageByExtraction, the dimension and location along that dimension for which data should be extracted. For generateImageDataForShape, dim is the dimensions of the image. newMriImageAsShapeOverlay takes this from the baseImage.

mask	An array of mode logical indicating which voxels are in the mask. Must have the same dimensions as the image.
level	A numeric value specifying the threshold level.
defaultValue	The value of the final image in voxels which are below threshold.
clearance	The number of voxels' clearance left around a trimmed image.
type	The shape type to generate. A "block" is a cubic region of the image; a "cross" is the central line of the cube in each dimension.
baseImage	The MriImage to use as a base for the overlay.
background	The voxel value outside the shape.
centre, width	The centre and width of the shape.
newDataType	The data type of the new image. If NULL, then the data type is the same as the source image.
fun	A function object, taking one or two numeric array parameters, as appropriate.
...	For newMriImageFromTemplate, further parameters to newMriImageMetadataFromTemplate . For newMriImageAsShapeOverlay, further parameters to generateImageDataForShape. And for newMriImageWithSimpleFunction and newMriImageWithBinaryFunction, further parameters to fun.

Details

All of these functions use data from arrays or MriImage objects to create a new MriImage object. newMriImageWithData is the basic function for creating an object from its constituents: an array of voxel values and an MriImageMetadata object. newMriImageFromTemplate takes an existing image, with its voxel data, and creates a new image, possibly with modifications to the metadata.

newMriImageByExtraction reduces the dimensionality of the source image by one, by extracting a single "line" of data along one dimension. (An array, rather than an MriImage object, is returned by extractDataFromMriImage.) newMriImageByMasking modifies the data by masking out unwanted voxels, and newMriImageByThresholding by thresholding. newMriImageByTrimming trims empty space from the edges of an image, reducing the dimensions of the image and thus avoiding the storage of lots of zeroes. newMriImageAsShapeOverlay creates an image which contains a simple shape. newMriImageWithSimpleFunction and newMriImageWithBinaryFunction modify the image data by applying an arbitrary function to it. Any function that can be applied to numeric arrays, and expects one or two arguments, respectively, is suitable for fun.

Value

An MriImage object.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[MriImage](#)

newMriImageWithDataRepresentation

Convert between image data storage conventions

Description

This function returns a version of its first argument using the specified data representation.

Usage

```
newMriImageWithDataRepresentation(image, representation = c("dense", "coordlist"))
```

Arguments

`image` An `MriImage` object.

`representation` A character string specifying the required data representation.

Value

A version of `image` with using the specified data representation, i.e. an array (with `representation="dense"`) or a `SparseArray` (with `representation="coordlist"`).

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[MriImage](#), [SparseArray](#), [array](#)

newSparseArrayWithData
Create a SparseArray object

Description

This function creates a [SparseArray](#) object from its constituent parts.

Usage

```
newSparseArrayWithData(data, coordinates, dims)
```

Arguments

data	A vector of (nonzero) array elements.
coordinates	A matrix with as many rows as data has elements, containing the coordinates of each nonzero element in the array.
dims	The dimensions of the array.

Value

A [SparseArray](#) object.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

nilObject *The nil object*

Description

The nil object is an empty object of class [SerialisableObject](#). It can be used as a placeholder where such an object of this class, or one of its subclasses, is required. It serialises to the empty list.

Usage

```
nilObject()  
is.nilObject(object)
```

Arguments

object Any object.

Value

nilObject returns the nil object. is.nilObject returns TRUE if its argument is identical to the nil object, or if it is equivalent in the sense of serialising to an identical result.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[SerialisableObject](#)

path manipulation *Functions for file name and path manipulation*

Description

Functions for expanding file paths, finding relative paths and ensuring that a file name has the required suffix.

Usage

```
ensureFileSuffix(fileName, suffix, strip = NULL)
expandFileName(fileName, base = getwd())
relativePath(path, referencePath)
```

Arguments

fileName A character vector of file names.

suffix A character vector of file suffixes, which will be recycled if shorter than fileName.

strip A character vector of suffixes to remove before appending suffix. The intended suffix does not need to be given here, as the function will not append it if the specified file name already has the correct suffix.

base If fileName is a relative path, this option gives the base directory which the path is relative to. If fileName is an absolute path, this argument is ignored.

path, referencePath Character strings representing file paths.

Value

The `ensureFileSuffix` function returns the specified file names with the requested suffixes appended. `expandFileName` returns the full path to the specified file name, collapsing "." elements if appropriate. `relativePath` returns the specified path, expressed relative to `referencePath`.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[path.expand](#) performs some of what `expandFileName` does.

printLabelledValues *Pretty print labelled information*

Description

This is a simple function to print a series of labels and associated data values, or key-value pairs.

Usage

```
printLabelledValues(labels, values, outputLevel = 0L$Info, leftJustify = FALSE)
```

Arguments

labels	A character vector of labels.
values	A character vector of values. Must have the same length as labels.
outputLevel	The output level to print the output to. See setOutputLevel , in the <code>reportr</code> package.
leftJustify	Logical value: if TRUE the labels will be left justified; otherwise they will be right justified.

Value

This function is called for its side effect.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[setOutputLevel](#) for the reportr output level system.

promote

Promote a vector to a single-column or single-row matrix

Description

The promote function promotes a vector argument to a single-column or single-row matrix. Matrix arguments are returned unmodified.

Usage

```
promote(x, byrow = FALSE)
```

Arguments

x	A vector or matrix.
byrow	Logical value: if TRUE, a vector will be promoted to a single-row matrix; otherwise a single-column matrix will result.

Value

A matrix version of the x argument.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[matrix](#)

SerialisableObject-class
Class "SerialisableObject"

Description

This reference class extends the standard "envRefClass" class, adding a function for simple serialisation of the data fields of an object, and one for finding all of the methods available for an object. A serialised object may be deserialised using the `deserialiseReferenceObject` function.

Extends

All reference classes extend and inherit methods from "envRefClass".

Class-Based Methods

`serialise(file)`: If the file argument is missing, serialises the object to a list containing the (named) fields of the object, with its "originalClass" attribute set to the class name of the original object. If the file argument is supplied, this object is saved to the corresponding file name using the standard `save` function.

`methods()`: Retrieve a character vector describing the methods available to the object.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

Examples

```
showClass("SerialisableObject")
```

Description

Rather than using R's `save` and `load` functions directly for reference objects, TractoR uses the `SerializableObject` class and these functions to save and load objects. The main difference is that this approach stores only the data in the object, and not the functions which operate on them. This helps backward compatibility when new member functions are added.

Usage

```
isDeserialisable(object, expectedClass = NULL)
deserialiseReferenceObject(file = NULL, object = NULL, raw = FALSE)
```

Arguments

<code>object</code>	A list object in (raw) serialised form. See Details.
<code>expectedClass</code>	A class name which the object is expected to inherit. Any class is acceptable if this parameter is NULL.
<code>file</code>	A file name to deserialise from.
<code>raw</code>	If TRUE, the raw serialised object is returned; otherwise the object is converted back to its original class.

Details

The `serialise` member function of the `SerializableObject` class can be used to create and/or `save` a version of an object which contains a hierarchical representation of the data embedded in it. These serialised objects are standard R lists, with an "originalClass" attribute describing the class of the original object. The functions listed above can be used to deserialise them.

Note that this should generally NOT be used as the primary mechanism for saving and loading `MriImage` objects. Saving to standard NIFTI/Analyze format is usually preferable, and can be done using `writeMriImageToFile`.

Value

`isDeserialisable` returns TRUE if the object is deserialisable and inherits from the specified class. `deserialiseReferenceObject` returns a raw or reconstituted object after deserialisation.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[SerialisableObject](#), [save](#), [load](#), [writeMriImageToFile](#).

sortDicomDirectory *Sort a directory of DICOM files into series*

Description

This function sorts a directory containing DICOM files into subdirectories by series number, DICOM tag (0x0020,0x0011), and/or subject name, DICOM tag (0x0010,0x0010). Each unique identifier, together with its description, will be used as the name for a new subdirectory of the specified top-level directory, and all relevant files will be copied into that subdirectory. Duplicate file names are disambiguated if necessary.

Usage

```
sortDicomDirectory(directory, deleteOriginals = FALSE, sortOn = "series")
```

Arguments

directory	A length-1 character vector giving the directory to search for DICOM files. Subdirectories will also be searched.
deleteOriginals	A single logical value. If TRUE, then the source files will be deleted after being copied to their new locations, making the operation a move rather than a copy. Nothing will be deleted if the copy fails.
sortOn	Either "series" or "subject", or both in the order desired. This will be the basis of the sort, which will be nested if both types are specified.

Value

This function is called for its side effect.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[newMriImageFromDicom](#) for reading DICOM files into an MriImage object.

SparseArray-class *Class "SparseArray"*

Description

This class represents an array with any number of dimensions, in which a significant proportion of entries are zero. The coordinates of nonzero entries are stored along with their values, with all remaining entries assumed to be zero. Methods are provided to index into the array in the standard way, using matrix or vector indices; and for coercing between SparseArray objects and standard (dense) arrays.

Extends

Class "[SerialisableObject](#)", directly.

All reference classes extend and inherit methods from "[envRefClass](#)".

Methods

[signature(x = "SparseArray"): ...
 [<- signature(x = "SparseArray"): ...

Fields

data: Object of class ANY.

coords: Object of class matrix.

dims: Object of class integer.

Class-Based Methods

initialize(...): Create a new object of this class.

getDimensions(): Retrieve an integer vector giving the dimensions of the array.

getDimensionality(): Retrieve an integer vector of length 1 giving the number of dimensions in the array.

getCoordinates(): Retrieve the matrix of coordinates of nonzero entries, with each row corresponding to a nonzero entry.

`getData()`: Retrieve the vector of nonzero data values.

`setCoordinatesAndData(coords, data)`: Update the coordinate matrix and data vector explicitly.

The following methods are inherited (from the corresponding class): `serialise` ("SerialisableObject"), `methods` ("SerialisableObject")

Note

Using the current “coordinate list” storage convention, there is rarely any data size benefit to storing an array in this form unless it is at least 75% sparse. Since `SparseArray` is a reference class, however, there may be some benefit in terms of avoiding duplication of the array on modification.

Indexing with logical or character vectors is currently not supported.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[SparseOrDenseArray](#) and [array](#). The implementation of indexing owes lot to the `slam` package.

Examples

```
showClass("SparseArray")
```

SparseOrDenseArray-class

Class "SparseOrDenseArray"

Description

This virtual class encapsulates either a standard R `array`, or an object of class `SparseArray`. It is used as the data field in `MriImage` objects.

Objects from the Class

A virtual Class: No objects may be created from it.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

Examples

```
showClass("SparseOrDenseArray")
```

threadSafeTempFile	<i>Obtain thread-safe temporary file names</i>
--------------------	--

Description

This function is a wrapper around `tempfile`, which creates temporary file names whose path contains the process ID of the calling process. This avoids clashes between threads created by functions such as `mclapply` (in the “parallel” package), which can easily occur with the standard `tempfile` function.

Usage

```
threadSafeTempFile(pattern = "file")
```

Arguments

`pattern` Character vector giving the initial part of each file name.

Value

A character vector of temporary file names. No files are actually created.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. Journal of Statistical Software 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also[tempfile](#)

vector functions	<i>Miscellaneous vector functions</i>
------------------	---------------------------------------

Description

These functions provide the (Euclidean) length of a vector, the vector cross product or angle between two vectors.

Usage

```
vectorLength(vector)
vectorCrossProduct(a, b)
angleBetweenVectors(v1, v2)
resolveVector(len, ...)
```

Arguments

vector, v1, v2	Numeric vectors of any length.
a, b	Numeric 3-vectors.
len	The expected length of the vector.
...	Elements of the vector, to be concatenated together.

Value

For `vectorLength`, the Euclidean norm or length of the specified vector, given by $\sqrt{\text{sum}(\text{vector}^2)}$. For `vectorCrossProduct`, the vector cross product of the two specified vectors; and for `angleBetweenVectors`, the angle (in radians) between the two specified vectors.

The `resolveVector` function concatenates the values given in `...`, and if the result is a vector of length `len` then it is returned. If not, `NULL` is returned.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[crossprod](#) for the matrix cross product.

visualisation	<i>Visualise MriImage objects</i>
---------------	-----------------------------------

Description

Visualise MriImage objects.

Usage

```
createCombinedGraphics(images, modes, colourScales, axes = 1:3, sliceLoc = NULL,
  device = c("internal", "png"), alphaImages = NULL,
  prefix = "image", zoomFactor = 1, filter = "Mitchell",
  windowLimits = NULL, clearance = NULL, nColumns = NULL)
```

```
createProjectionGraphic(image, axis, device = c("internal", "png"), colourScale = 1,
  add = FALSE, file = NULL, zoomFactor = 1, filter = "Mitchell",
  windowLimits = NULL)
```

```
createSliceGraphic(image, x = NA, y = NA, z = NA, device = c("internal", "png"),
  colourScale = 1, add = FALSE, file = NULL, zoomFactor = 1,
  filter = "Mitchell", windowLimits = NULL)
```

```
createContactSheetGraphic(image, axis, device = c("internal", "png"), colourScale = 1,
  add = FALSE, file = NULL, zoomFactor = 1, filter = "Mitchell",
  windowLimits = NULL, clearance = NULL, nColumns = NULL)
```

Arguments

images	A list of MriImage objects.
image	A single MriImage object.
modes	A character vector of the same length as images, each element being "slice" or "projection" (or abbreviations), indicating which type of visualisation should be applied to each image.
colourScale, colourScales	A single colour scale definition, or a list in the plural case. See Details.
axis, axes	A vector of axes along which slice/projection images should be created. 1 is left-right, 2 is anterior-posterior, 3 is superior-inferior.
x, y, z	Integer vectors, each of length 1. Exactly one of these must be specified to indicate the plane of interest.
sliceLoc	Like x, y and z, except that a point in 3 dimensions is specified. Must not be NA for each axis requested.
device	Either "internal" for display on the default graphics device, or "png" for creating PNG format image file(s). Abbreviations are fine.
alphaImages	A list of MriImage objects to be used as transparency masks. Must be the same length as images if not NULL. NULL values in the list indicate no mask.

prefix, file	A file name or prefix (to which "axial", "coronal" or "sagittal" will be added) to be used when device is "png".
zoomFactor	Factor by which to enlarge the image. Applies only when device is "png".
filter	Interpolation filter to be used by ImageMagick.
windowLimits	Numeric vector of length 2 giving the limits of the colour scale, or NULL for limits matching the range of the image data. Passed as the zlim argument to image .
clearance	Number of voxels' clearance to leave around each slice image in the contact sheet. Passed to newMriImageByTrimming .
nColumns	Number of slices per row in the contact sheet grid. If NULL, the function will aim for a square grid.
add	Overlay the graphic on a previous one. Used only when device is "internal".

Details

These functions create 2D visualisations of 3D images by slicing or maximum intensity projection. Colour scales can be specified in any of three ways. Firstly, by a single number, representing a predefined colour scale. Currently valid values are 1 (greyscale, black background), 2 (red to yellow heat scale, red background), 3 (blue to red rainbow scale, blue background) and 4 (blue to white to red diverging scale, white background). Secondly, a single colour name can be given (see [colours](#)); in this case the background will be black. This is useful for binary images. Thirdly and most flexibly, a list with two named elements can be given: `colours`, a vector of colours representing the colour scale, perhaps created using [rgb](#); and `background`, a single colour representing the background.

NB: When the device option is set to "png", ImageMagick is required by these functions.

Value

These functions are called for their side effects.

Author(s)

Jon Clayden

References

Please cite the following reference when using TractoR in your work:

J.D. Clayden, S. Muñoz Maniega, A.J. Storkey, M.D. King, M.E. Bastin & C.A. Clark (2011). TractoR: Magnetic resonance imaging and tractography with R. *Journal of Statistical Software* 44(8):1-18. <http://www.jstatsoft.org/v44/i08/>.

See Also

[image](#), [colours](#), [rgb](#)

Index

*Topic **classes**

- DicomMetadata-class, 3
- MriImage-class, 8
- MriImageMetadata-class, 10
- SerializableObject-class, 25
- SparseArray-class, 28
- SparseOrDenseArray-class, 29

*Topic **datasets**

- dictionary, 5

- [,MriImage-method (MriImage-class), 8
- [,SparseArray-method
(SparseArray-class), 28
- [.MriImage (MriImage-class), 8
- [<-,MriImage-method (MriImage-class), 8
- [<-,SparseArray-method
(SparseArray-class), 28
- [<-.MriImage (MriImage-class), 8

- all.equal, 5, 6

- angleBetweenVectors (vector functions),
31

- array, 8, 20, 29

- as.array.MriImage (MriImage-class), 8

- as.array.SparseArray
(SparseArray-class), 28

- colours, 33

- copyImageFiles (newMriImageFromFile), 15

- createCombinedGraphics (visualisation),
32

- createContactSheetGraphic
(visualisation), 32

- createNeighbourhoodInfo
(neighbourhoodInfo), 12

- createProjectionGraphic
(visualisation), 32

- createSliceGraphic (visualisation), 32

- crossprod, 31

- data types, 2

- deserialiseReferenceObject, 25

- deserialiseReferenceObject
(serialisation), 26

- DicomMetadata, 13, 15

- DicomMetadata (DicomMetadata-class), 3

- DicomMetadata-class, 3

- dictionary, 5, 13

- dim.MriImage (MriImage-class), 8

- dim.SparseArray (SparseArray-class), 28

- ensureFileSuffix (path manipulation), 22

- envRefClass, 3, 8, 10, 25, 28

- equivalent, 5

- execute, 6

- expandFileName (path manipulation), 22

- extractDataFromMriImage
(newMriImageWithData), 18

- file.copy, 16

- file.symLink, 16

- generateImageDataForShape
(newMriImageWithData), 18

- getDataByNiftiCode, 11, 15, 17, 18

- getDataByNiftiCode (data types), 2

- identifyImageFileNames
(newMriImageFromFile), 15

- image, 33

- imageFileExists (newMriImageFromFile),
15

- implode, 7

- is.nilObject (nilObject), 21

- isDeserialisable (serialisation), 26

- load, 26, 27

- locateExecutable (execute), 6

- Math, 8, 10

- Math.MriImage (MriImage-class), 8

- matrix, 24

- MriImage, 3, 10, 13, 15, 17, 18, 20, 26, 29, 32
- MriImage (MriImage-class), 8
- MriImage-class, 8
- MriImageMetadata, 8–10, 15, 17, 18
- MriImageMetadata
 - (MriImageMetadata-class), 10
- MriImageMetadata-class, 10

- neighbourhoodInfo, 12
- newDicomMetadataFromFile, 13
- newMriImageAsShapeOverlay
 - (newMriImageWithData), 18
- newMriImageByExtraction
 - (newMriImageWithData), 18
- newMriImageByMasking
 - (newMriImageWithData), 18
- newMriImageByThresholding
 - (newMriImageWithData), 18
- newMriImageByTrimming, 33
- newMriImageByTrimming
 - (newMriImageWithData), 18
- newMriImageFromDicom, 13, 14, 28
- newMriImageFromDicomDirectory
 - (newMriImageFromDicom), 14
- newMriImageFromDicomMetadata
 - (newMriImageFromDicom), 14
- newMriImageFromFile, 15
- newMriImageFromTemplate
 - (newMriImageWithData), 18
- newMriImageMetadataFromDicom
 - (newMriImageFromDicom), 14
- newMriImageMetadataFromDicomMetadata
 - (newMriImageFromDicom), 14
- newMriImageMetadataFromFile
 - (newMriImageFromFile), 15
- newMriImageMetadataFromTemplate, 17, 19
- newMriImageWithBinaryFunction, 10
- newMriImageWithBinaryFunction
 - (newMriImageWithData), 18
- newMriImageWithData, 18
- newMriImageWithDataRepresentation, 20
- newMriImageWithSimpleFunction, 10
- newMriImageWithSimpleFunction
 - (newMriImageWithData), 18
- newSparseArrayWithData, 21
- nilObject, 21

- Ops, 8, 10
- Ops.MriImage (MriImage-class), 8

- options, 16

- paste, 8
- path manipulation, 22
- path.expand, 23
- printLabelledValues, 23
- promote, 24

- relativePath (path manipulation), 22
- removeImageFilesWithName
 - (newMriImageFromFile), 15
- resolveVector (vector functions), 31
- rgb, 33

- save, 25–27
- SerializableObject, 3, 8, 10, 21, 22, 26–28
- SerializableObject
 - (SerializableObject-class), 25
- SerializableObject-class, 25
- serialisation, 26
- setOutputLevel, 23, 24
- sortDicomDirectory, 15, 27
- SparseArray, 20, 21, 29
- SparseArray (SparseArray-class), 28
- SparseArray-class, 28
- SparseOrDenseArray, 29
- SparseOrDenseArray
 - (SparseOrDenseArray-class), 29
- SparseOrDenseArray-class, 29
- Summary, 8, 10
- Summary.MriImage (MriImage-class), 8
- symlinkImageFiles
 - (newMriImageFromFile), 15
- system, 6, 7

- tempfile, 30, 31
- threadSafeTempFile, 30

- unlink, 16, 17

- vector functions, 31
- vectorCrossProduct (vector functions), 31
- vectorLength (vector functions), 31
- visualisation, 32

- writeMriImageToFile, 26, 27
- writeMriImageToFile
 - (newMriImageFromFile), 15