Package 'qMRI'

July 12, 2019

Type Package

Title Methods for Quantitative Magnetic Resonance Imaging ('qMRI')

Version 1.0.1

Date 2019-07-12

Maintainer Karsten Tabelow <karsten.tabelow@wias-berlin.de>

Depends R (>= 3.1), awsMethods (>= 1.0)

Imports oro.nifti (>= 0.9), stringr, dti (>= 1.4), aws (>= 2.2), adimpro (>= 0.9)

LazyData TRUE

Description Implementation of methods for estimation of quantitative maps from Multi-Parameter Mapping (MPM) acquisitions (Weiskopf et al. (2013) <doi:10.3389/fnins.2013.00095>) including adaptive smoothing methods in the framework of the ESTATICS model (Estimating the apparent transverse relaxation time (R2*) from images with different contrasts, Weiskopf et al. (2014) <doi:10.3389/fnins.2014.00278>). The smoothing method is described in Mohammadi et al. (2017). <doi:10.20347/WIAS.PREPRINT.2432>.

License GPL (>= 2)

URL http://www.wias-berlin.de/research/ats/imaging/

Suggests covr, testthat, knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.1.1

NeedsCompilation yes

Author Joerg Polzehl [aut], Karsten Tabelow [aut, cre], WIAS Berlin [cph, fnd]

Repository CRAN

Date/Publication 2019-07-12 14:20:03 UTC

R topics documented:

`qMRI-package` *Methods for Quantitative Magnetic Resonance Imaging ('qMRI')*

Description

Implementation of methods for estimation of quantitative maps from Multi-Parameter Mapping (MPM) acquisitions (Weiskopf et al. (2013) <doi:10.3389/fnins.2013.00095>) including adaptive smoothing methods in the framework of the ESTATICS model (Estimating the apparent transverse relaxation time (R2*) from images with different contrasts, Weiskopf et al. (2014) <doi:10.3389/fnins.2014.00278>). The smoothing method is described in Mohammadi et al. (2017). <doi:10.20347/WIAS.PREPRINT.2432>.

Details

The DESCRIPTION file:

Package:	qMRI
Type:	Package
Title:	Methods for Quantitative Magnetic Resonance Imaging ('qMRI')
Version:	1.0.1
Date:	2019-07-12
Authors@R:	c(person("Joerg", "Polzehl", role = c("aut"), email = "joerg.polzehl@wias-berlin.de"), person("Karsten", "
Maintainer:	Karsten Tabelow <karsten.tabelow@wias-berlin.de>
Depends:	R (>= 3.1), awsMethods (>= 1.0)
Imports:	oro.nifti (>= 0.9), stringr, dti (>= 1.4), aws (>= 2.2), adimpro (>= 0.9)
LazyData:	TRUE
Description:	Implementation of methods for estimation of quantitative maps from Multi-Parameter Mapping (MPM) ac
License:	GPL (>= 2)
Copyright:	This package is Copyright (C) 2015-2019 Weierstrass Institute for Applied Analysis and Stochastics.
URL:	http://www.wias-berlin.de/research/ats/imaging/
Suggests:	covr, testthat, knitr, rmarkdown
VignetteBuilder:	knitr
RoxygenNote:	6.1.1
Author:	Joerg Polzehl [aut], Karsten Tabelow [aut, cre], WIAS Berlin [cph, fnd]

Index of help topics:

Author(s)

Karsten Tabelow <tabelow@wias-berlin.de>
J\"org Polzehl <polzehl@wias-berlin.de>

Maintainer: Karsten Tabelow <karsten.tabelow@wias-berlin.de>

References

Weiskopf, N.; Suckling, J.; Williams, G.; Correia, M. M.; Inkster, B.; Tait, R.; Ooi, C.; Bullmore, E. T. & Lutti, A. Quantitative multi-parameter mapping of R1, PD(*), MT, and R2(*) at 3T: a multi-center validation. Front Neurosci, Wellcome Trust Centre for Neuroimaging, UCL Institute of Neurology, University College London, UK., 2013, 7, 95

See Also

aws

Examples

```
dataDir <- system.file("extdata",package="qMRI")
#
#  set file names for T1w, MTw and PDw images
#
t1Names <- paste0("t1w_",1:8,".nii.gz")
mtNames <- paste0("mtw_",1:6,".nii.gz")
pdNames <- paste0("pdw_",1:8,".nii.gz")
t1Files <- file.path(dataDir, t1Names)
mtFiles <- file.path(dataDir, mtNames)
pdFiles <- file.path(dataDir, pdNames)
```

```
#
#  file names of mask and B1 field map
#
B1File <- file.path(dataDir, "B1map.nii.gz")
maskFile <- file.path(dataDir, "mask.nii.gz")
#
#  Acquisition parameters (TE, TR, Flip Angle) for T1w, MTw and PDw images
#
TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
TR <- rep(25, 22)
FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
#
#   read MPM example data
#
library(qMRI)
mpm <- readMPMData(t1Files, pdFiles, mtFiles,
                   maskFile, TR = TR, TE = TE,
                   FA = FA, verbose = FALSE)
#
#  Estimate Parameters in the ESTATICS model
#
modelMPM <- estimateESTATICS(mpm, method = "NLR")
#
#  smooth maps of ESTATICS Parameters
#
setCores(2)
modelMPMQLsp1 <- smoothESTATICS(modelMPM,
                                kstar = 16,
                                alpha = 0.004,
                                patchsize=1,
                                verbose = TRUE)
#
# limit calculations to voxel in the central coronal slice
# to reduce execution time of the example
#
mpm$mask[,c(1:10,12:21),] <- FALSE
#
# Alternatively using Quasi-Likelihood
#
sigma <- 50
modelMPMQL <- estimateESTATICS(mpm, method = "QL",
                  sigma = array(sigma,mpm$sdim), L = 1)

#
#  resulting ESTATICS parameter maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
oldpar <- par(mfrow=c(2,4),mar=c(3,3,3,1),mgp=c(2,1,0))
pnames <- c("T1","MT","PD","R2star")
for(i in 1:4){
```

```
   rimage(modelMPMQL$modelCoeff[i,,11,])
   title(pnames[i])
   }
for(i in 1:4){
   rimage(modelMPMQLsp1$modelCoeff[i,,11,])
   title(paste("smoothed",pnames[i]))
   }
}
#
#  Compute quantitative maps (R1, R2star, PD, MT)
#
qMRIQLMaps <- calculateQI(modelMPMQL,
                        b1File = B1File,
                        TR2 = 3.4)
qMRIQLSmoothedp1Maps <- calculateQI(modelMPMQLsp1,
                                    b1File = B1File,
                                    TR2 = 3.4)
#
#  resulting quantitative maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
par(mfrow=c(2,4),mar=c(3,3,3,1),mgp=c(2,1,0))
nmaps <- c("R1","R2star","PD","MT")
qmap <- extract(qMRIQLMaps,nmaps)
for (i in 1:4) rimage(qmap[[i]][,11,],main=nmaps[i])
qmap <- extract(qMRIQLSmoothedp1Maps,nmaps)
for (i in 1:4) rimage(qmap[[i]][,11,],main=paste("Smoothed",nmaps[i]))
}
par(oldpar)
```

`calculateQI` *Obtain quantitative maps from estimated ESTATICS parameters.*

Description

Quantitaive imaging parameters are calculated from the estimated parameters in the ESTATICS model. This involves a correction for magnetic field inhomogeneities if the information is provided in argument `b1File` and use of a second of a second recovery delay TR2 in case of Dual-Exitation FLASH measurements (Helms 2008).

Usage

```
calculateQI(mpmESTATICSModel, b1File = NULL, TR2 = 0, verbose = TRUE)
```

Arguments

`mpmESTATICSModel`
: Object of class 'ESTATICSModel' as returned from function `estimateESTATICS`.

b1File	(optional) Name of a file containing a B1-field inhomogeneity map (.nii)
TR2	second recovery delay TR2 in case of Dual-Exitation FLASH measurements.
verbose	logical: Monitor process.

Value

List with components

b1Map	b1Map
R1	Estimated map of R1
R2star	Estimated map of R2star
PD	Estimated map of PD
MT	Estimated map of delta (if MT-series was used)
model	Type of ESTATICS model used
t1Files	filenames T1
mtFiles	filenames MT
pdFiles	filenames PD
mask	brainmask

and class-attribute 'qMaps' .

Author(s)

Karsten Tabelow <tabelow@wias-berlin.de>
J\"org Polzehl <polzehl@wias-berlin.de>

References

Helms, G.; Dathe, H.; Kallenberg, K. & Dechent, P. High-Resolution Maps of Magnetization Transfer with Inherent Correction for RF Inhomogeneity and T1 Relaxation Obtained from 3D FLASH MRI Magn. Res. Med., 2008, 60, 1396-1407

Weiskopf, N.; Suckling, J.; Williams, G.; Correia, M. M.; Inkster, B.; Tait, R.; Ooi, C.; Bullmore, E. T. & Lutti, A. Quantitative multi-parameter mapping of R1, PD(*), MT, and R2(*) at 3T: a multi-center validation. Front Neurosci, Wellcome Trust Centre for Neuroimaging, UCL Institute of Neurology, University College London, UK., 2013, 7, 95

See Also

readMPMData, estimateESTATICS, smoothESTATICS, writeESTATICS, awslsigmc

Examples

```
dataDir <- system.file("extdata",package="qMRI")
#
#  set file names for T1w, MTw and PDw images
#
t1Names <- paste0("t1w_",1:8,".nii.gz")
mtNames <- paste0("mtw_",1:6,".nii.gz")
pdNames <- paste0("pdw_",1:8,".nii.gz")
t1Files <- file.path(dataDir, t1Names)
mtFiles <- file.path(dataDir, mtNames)
pdFiles <- file.path(dataDir, pdNames)
#
#  file names of mask and B1 field map
#
B1File <- file.path(dataDir, "B1map.nii.gz")
maskFile <- file.path(dataDir, "mask.nii.gz")
#
#  Acquisition parameters (TE, TR, Flip Angle) for T1w, MTw and PDw images
#
TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
TR <- rep(25, 22)
FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
#
#   read MPM example data
#
library(qMRI)
mpm <- readMPMData(t1Files, pdFiles, mtFiles,
                   maskFile, TR = TR, TE = TE,
                   FA = FA, verbose = FALSE)
#
# limit calculations to voxel in the central coronal slice
# to reduce execution time of the example
#
mpm$mask[,c(1:10,12:21),] <- FALSE
#
#  Estimate Parameters in the ESTATICS model
#
modelMPM <- estimateESTATICS(mpm, method = "NLR")
#
#  resulting ESTATICS parameter maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
oldpar <- par(mfrow=c(2,2),mar=c(3,3,3,1),mgp=c(2,1,0))
pnames <- c("T1","MT","PD","R2star")
for(i in 1:4){
   rimage(modelMPM$modelCoeff[i,,11,])
   title(pnames[i])
   }
}
```

```
#
#  Compute quantitative maps (R1, R2star, PD, MT)
#
qMRIMaps <- calculateQI(modelMPM,
                        b1File = B1File,
                        TR2 = 3.4)
#
#  resulting quantitative maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
par(mfrow=c(2,2),mar=c(3,3,3,1),mgp=c(2,1,0))
nmaps <- c("R1","R2star","PD","MT")
qmap <- extract(qMRIMaps,nmaps)
for (i in 1:4) rimage(qmap[[i]][,11,],main=nmaps[i])
}
par(oldpar)
```

`colMT` *MT map color scheme*

Description

Color map implementing the color scheme for MT maps. This is the plasma scale from Matplotlib (pyplot) generated by function plasma from package **viridisLite**.

Usage

```
colMT
```

Format

A vector with 256 RGB color values.

`estimateESTATICS` *Estimate parameters in the ESTATICS model.*

Description

Evaluation of the ESTATICS model (Weisskopf (2013) using nonlinear least squares regression and a quasi-likelihood approach assuming a noncentral chi- or a Rician distribuion for the data. The latter should be preferred in case of low SNR (high resolution) data to avoid biased parameter estimates. Quasi-likelihood estimation requires a specification of the scale parameter sigma of the data distribution.

Usage

```
estimateESTATICS(mpmdata, TEScale = 100, dataScale = 1000, method = c("NLR", "QL"),
                 sigma = NULL, L = 1, maxR2star = 50, verbose = TRUE)
```

Arguments

mpmdata	Object of class MPMData as created by readMPMData.
TEScale	scale factor for TE (used for improved numerical stability)
dataScale	scale factor for image intensities (used for improved numerical stability)
method	either "NLR" or "QL". Specifies non-linear regression or quasi-likelihood.
sigma	scale parameter sigma of signal distribution (either a scalar or a 3D array). (only needed in case of method="QL".)
L	effective number of receiver coils (2*L is degrees of freedom of the signal distribution). L=1 for Rician distribution. (only needed in case of method="QL".)
maxR2star	maximum value allowed for the R2star parameter in the ESTATICS model.
verbose	logical: Monitor process.

Value

list with components

modelCoeff	Estimated parameter maps
invCov	map of inverse covariance matrices
rsigma	map of residual standard deviations
isConv	convergence indicator map
isThresh	logical map indicating where R2star==maxR2star.
sdim	image dimension
nFiles	number of images
t1Files	vector of T1 filenames
pdFiles	vector of PD filenames
mtFiles	vector of MT filenames
model	model used (depends on specification of MT files)
maskFile	filename of brain mask
mask	brain mask
sigma	sigma
L	L
TR	TR values
TE	TE values
FA	Flip angles (FA)
TEScale	TEScale
dataScale	dataScale

and class-attribute 'ESTATICSModel'

Author(s)

Karsten Tabelow <tabelow@wias-berlin.de>
J\"org Polzehl <polzehl@wias-berlin.de>

References

Weiskopf, N.; Suckling, J.; Williams, G.; Correia, M. M.; Inkster, B.; Tait, R.; Ooi, C.; Bullmore, E. T. & Lutti, A. Quantitative multi-parameter mapping of R1, PD(*), MT, and R2(*) at 3T: a multi-center validation. Front Neurosci, Wellcome Trust Centre for Neuroimaging, UCL Institute of Neurology, University College London, UK., 2013, 7, 95

See Also

readMPMData, calculateQI, smoothESTATICS, writeESTATICS, awslsigmc

Examples

```
dataDir <- system.file("extdata",package="qMRI")
#
#  set file names for T1w, MTw and PDw images
#
t1Names <- paste0("t1w_",1:8,".nii.gz")
mtNames <- paste0("mtw_",1:6,".nii.gz")
pdNames <- paste0("pdw_",1:8,".nii.gz")
t1Files <- file.path(dataDir, t1Names)
mtFiles <- file.path(dataDir, mtNames)
pdFiles <- file.path(dataDir, pdNames)
#
#  file names of mask and B1 field map
#
B1File <- file.path(dataDir, "B1map.nii.gz")
maskFile <- file.path(dataDir, "mask.nii.gz")
#
#  Acquisition parameters (TE, TR, Flip Angle) for T1w, MTw and PDw images
#
TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
TR <- rep(25, 22)
FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
#
#   read MPM example data
#
library(qMRI)
mpm <- readMPMData(t1Files, pdFiles, mtFiles,
                   maskFile, TR = TR, TE = TE,
                   FA = FA, verbose = FALSE)
#
# limit calculations to voxel in the central coronal slice
# to reduce execution time of the example
```

```
#
mpm$mask[,c(1:10,12:21),] <- FALSE
#
#  Estimate Parameters in the ESTATICS model
#
modelMPM <- estimateESTATICS(mpm, method = "NLR")
# Alternatively using Quasi-Likelihood
sigma <- 50
modelMPMQL <- estimateESTATICS(mpm, method = "QL",
                  sigma = array(sigma,mpm$sdim), L = 1)
```

`extract-methods` *Methods to extract information from objects of class* `"MPMData"`*,* `"ESTATICSModel"`*,* `"sESTATICSModel"` *and* `"qMaps"`.

Description

The methods extract and/or compute specified statistics from object of class `"MPMData"`, `"ESTATICSModel"`, `"sESTATICSModel"` and `"qMaps"`.

Usage

```
## S3 method for class 'MPMData'
extract(x, what, ...)
## S3 method for class 'ESTATICSModel'
extract(x, what, ...)
## S3 method for class 'qMaps'
extract(x, what, ...)
```

Arguments

`x`	object of class `"MPMData"`, `"ESTATICSModel"`, `"sESTATICSModel"` or `"qMaps"`.
`what`	Character vector of of names of statistics to extract. See Methods Section for details.
`...`	additional parameters, currently unused.

Value

A list with components carrying the names of the options specified in argument `what`.

Methods

class(x) = "ANY" Returns a warning for `extract`

class(x) = "MPMData" Depending the occurence of names in `what` a list with the specified components is returned

- "ddata"mpm data

- "sdim" dimension of image cube
- "nFiles" number of images / image files
- "t1Files" character - filenames of t1Files
- "pdFiles" character - filenames of pdFiles
- "mtFiles" character - filenames of mtFiles
- "model" Number of the ESTATICS model that can be used
- "maskFile" character - filenames of maskFile
- "mask" mask
- "TR" vector of TR values
- "TE" vector of TE values
- "FA" vector of FA values

class(x) = "ESTATICSModel" Depending the occurence of names in `what` a list with the specified components is returned

- "modelCoeff"Estimated parameter maps
- "invCov" map of inverse covariance matrices
- "rsigma"map of residual standard deviations
- "isConv" convergence indicator map
- "isThresh" logical map indicating where `R2star==maxR2star`.
- "sdim" image dimension
- "nFiles" number of images
- "t1Files" vector of T1 filenames
- "pdFiles" vector of PD filenames
- "mtFiles" vector of MT filenames
- "model" model used (depends on specification of MT files)
- "maskFile" filename of brain mask
- "mask" brain mask
- "sigma" sigma
- "L" L
- "TR" TR values
- "TE" TE values
- "FA" Flip angles (FA)
- "TEScale" TEScale
- "dataScale" dataScale

class(x) = "sESTATICSModel" Depending the occurence of names in `what` a list with the specified components is returned

- "modelCoeff"Estimated parameter maps
- "invCov" map of inverse covariance matrices
- "rsigma" map of residual standard deviations
- "isConv" convergence indicator map
- "bi" Sum of weights map from AWS/PAWS
- "smoothPar" smooting parameters used in AWS/PAWS
- "smoothedData" smoothed mpmData

- "isThresh" logical map indicating where `R2star==maxR2star`.
- "sdim" image dimension
- "nFiles" number of images
- "t1Files" vector of T1 filenames
- "pdFiles" vector of PD filenames
- "mtFiles" vector of MT filenames
- "model" model used (depends on specification of MT files)
- "maskFile" filename of brain mask
- "mask" brain mask
- "sigma" sigma
- "L" L
- "TR" TR values
- "TE" TE values
- "FA" Flip angles (FA)
- "TEScale" TEScale
- "dataScale" dataScale

class(x) = "qMaps" Depending the occurence of names in `what` a list with the specified components is returned

- b1Map b1Map
- R1 Estimated map of R1
- R2star Estimated map of R2star
- PD Estimated map of PD
- MT Estimated map of delta (if MT-series was used)
- model Type of ESTATICS model used
- t1Files filenames T1
- mtFiles filenames MT
- pdFiles filenames PD
- mask brainmask

Author(s)

Karsten Tabelow <tabelow@wias-berlin.de>
J\"org Polzehl <polzehl@wias-berlin.de>

Examples

```
dataDir <- system.file("extdata",package="qMRI")
#
#  set file names for T1w, MTw and PDw images
#
t1Names <- paste0("t1w_",1:8,".nii.gz")
mtNames <- paste0("mtw_",1:6,".nii.gz")
pdNames <- paste0("pdw_",1:8,".nii.gz")
t1Files <- file.path(dataDir, t1Names)
mtFiles <- file.path(dataDir, mtNames)
```

```
pdFiles <- file.path(dataDir, pdNames)
#
#  file names of mask and B1 field map
#
B1File <- file.path(dataDir, "B1map.nii.gz")
maskFile <- file.path(dataDir, "mask.nii.gz")
#
#  Acquisition parameters (TE, TR, Flip Angle) for T1w, MTw and PDw images
#
TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
TR <- rep(25, 22)
FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
#
#   read MPM example data
#
library(qMRI)
mpm <- readMPMData(t1Files, pdFiles, mtFiles,
                   maskFile, TR = TR, TE = TE,
                   FA = FA, verbose = FALSE)
#
#  display some data
#
data <- extract(mpm,"ddata")
if(require(adimpro)){
rimage.options(ylab = "z")
oldpar <- par(mfrow=c(1,3),mar=c(3,3,3,1),mgp=c(2,1,0))
rimage(data[1,,11,], main="first T1w image")
rimage(data[9,,11,], main="first MTw image")
rimage(data[15,,11,], main="first PDw image")
}
#
# limit calculations to voxel in the central coronal slice
# to reduce execution time of the example
#
mpm$mask[,c(1:10,12:21),] <- FALSE
#
#  Estimate Parameters in the ESTATICS model
#
modelMPM <- estimateESTATICS(mpm, method = "NLR")
#
#  Parameter maps and residual standard deviation
#
z <- extract(modelMPM,c("rsigma","modelCoeff"))
if(require(adimpro)){
rimage.options(ylab = "z")
par(mfrow=c(1,5),mar=c(3,3,3,1),mgp=c(2,1,0))
rimage(z$modelCoeff[1,,11,], main="S_T1")
rimage(z$modelCoeff[2,,11,], main="S_MT")
rimage(z$modelCoeff[3,,11,], main="S_PD")
rimage(z$modelCoeff[4,,11,], main="R2star")
rimage(z$rsigma[,11,], main="Residual sd")
```

```
}
#
#  Compute quantitative maps (R1, R2star, PD, MT)
#
qMRIMaps <- calculateQI(modelMPM,
                        b1File = B1File,
                        TR2 = 3.4)
#
#  resulting quantitative maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
par(mfrow=c(2,4),mar=c(3,3,3,1),mgp=c(2,1,0))
nmaps <- c("R1","R2star","PD","MT")
qmap <- extract(qMRIMaps,nmaps)
for (i in 1:4) rimage(qmap[[i]][,11,],main=nmaps[i])
}
par(oldpar)
```

`readMPMData` *Read experimental Multi-Parameter Mapping (MPM) data.*

Description

The function reads data generated in Multimodal Parameter Mapping (MPM) experiments.

Usage

```
readMPMData(t1Files = NULL, pdFiles = NULL, mtFiles = NULL, maskFile = NULL,
            TR = NULL, TE = NULL, FA = NULL, verbose = TRUE)
```

Arguments

`t1Files`	Vector of filenames corresponding to T1 weighted images (in Nifti-Format) with varying TE
`pdFiles`	Vector of filenames corresponding to PD weighted images (in Nifti-Format) with varying TE
`mtFiles`	optional Vector of filenames corresponding to MT weighted images (in Nifti-Format) with varying TE
`maskFile`	optional filename for mask (in Nifti-Format)
`TR`	optional numeric TR vector, if omitted information is extracted from .nii files if possible
`TE`	optional numeric TE vector, if omitted information is extracted from .nii files if possible
`FA`	optional numeric FA (flip-angle) vector, if omitted information is extracted from .nii files if possible
`verbose`	logical - provide information on progress

Value

List with components

ddata	mpm data
sdim	dimension of image cube
nFiles	number of images / image files
t1Files	character - filenames of t1Files
pdFiles	character - filenames of pdFiles
mtFiles	character - filenames of mtFiles
model	Number of the ESTATICS model that can be used
maskFile	character - filenames of maskFile
mask	mask
TR	vector of TR values
TE	vector of TE values
FA	vector of FA values

and class-attribute 'mpmData'

Author(s)

Karsten Tabelow <tabelow@wias-berlin.de>
J\"org Polzehl <polzehl@wias-berlin.de>

References

Weiskopf, N.; Suckling, J.; Williams, G.; Correia, M. M.; Inkster, B.; Tait, R.; Ooi, C.; Bullmore, E. T. & Lutti, A. Quantitative multi-parameter mapping of R1, PD(*), MT, and R2(*) at 3T: a multi-center validation. Front Neurosci, Wellcome Trust Centre for Neuroimaging, UCL Institute of Neurology, University College London, UK., 2013, 7, 95

See Also

estimateESTATICS, calculateQI, smoothESTATICS, writeESTATICS, awslsigmc

Examples

```
dataDir <- system.file("extdata",package="qMRI")
#
#  set file names for T1w, MTw and PDw images
#
t1Names <- paste0("t1w_",1:8,".nii.gz")
mtNames <- paste0("mtw_",1:6,".nii.gz")
pdNames <- paste0("pdw_",1:8,".nii.gz")
t1Files <- file.path(dataDir, t1Names)
mtFiles <- file.path(dataDir, mtNames)
pdFiles <- file.path(dataDir, pdNames)
#
```

```
#  file names of mask and B1 field map
#
B1File <- file.path(dataDir, "B1map.nii.gz")
maskFile <- file.path(dataDir, "mask.nii.gz")
#
#  Acquisition parameters (TE, TR, Flip Angle) for T1w, MTw and PDw images
#
TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
TR <- rep(25, 22)
FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
#
#   read MPM example data
#
library(qMRI)
mpm <- readMPMData(t1Files, pdFiles, mtFiles,
                   maskFile, TR = TR, TE = TE,
                   FA = FA, verbose = FALSE)
```

`smoothESTATICS` *Adaptive smoothing of ESTATICS parameters and MPM data*

Description

Performs adaptive smoothing of parameter maps in the ESTATICS model and if `mpmData` is specified these data. Implements both vectorized variants of the Adaptive Weights Smoothing (AWS, Polzehl and Spokoiny (2006)) and patchwise AWS (PAWS, Polzehl et al (2018)) algorithms with weighting schemes determined by the estimated parameter maps and their covariances.

Usage

```
smoothESTATICS(mpmESTATICSModel, mpmData = NULL, kstar = 16, alpha = 0.025,
               patchsize = 0, wghts = NULL, verbose = TRUE)
```

Arguments

`mpmESTATICSModel`	Object of class 'ESTATICSModel' as returned from function `estimateESTATICS`.
`mpmData`	(optional) Object of class MPMData as created by `readMPMData` from which the parameter maps were obtained.
`kstar`	Maximum number of steps.
`alpha`	specifies the scale parameter for the adaptation criterion. smaller values are more restrictive.
`patchsize`	Patchsize in PAWS, 0 corresponds to AWS, alternative values are 1 and 2.
`wghts`	(optional) voxel size if measurments are not isotropic.
`verbose`	logical - provide information on progress

Value

list with components

`modelCoeff`	Estimated parameter maps
`invCov`	map of inverse covariance matrices
`isConv`	convergence indicator map
`bi`	Sum of weights map from AWS/PAWS
`smoothPar`	smooting parameters used in AWS/PAWS
`smoothedData`	smoothed mpmData
`sdim`	image dimension
`nFiles`	number of images
`t1Files`	vector of T1 filenames
`pdFiles`	vector of PD filenames
`mtFiles`	vector of MT filenames
`model`	model used (depends on specification of MT files)
`maskFile`	filename of brain mask
`mask`	brain mask
`sigma`	sigma
`L`	L
`TR`	TR values
`TE`	TE values
`FA`	Flip angles (FA)
`TEScale`	TEScale
`dataScale`	dataScale

and class-attribute 'sESTATICSModel'

Author(s)

Karsten Tabelow <tabelow@wias-berlin.de>
J\"org Polzehl <polzehl@wias-berlin.de>

References

Joerg Polzehl, Vladimir Spokoiny, Propagation-separation approach for local likelihood estimation, Probab. Theory Related Fields 135 (3), (2006) , pp. 335–362.

Joerg Polzehl, Kostas Papafitsorus, Karsten Tabelow (2018). Patch-wise adaptive weights smoothing. WIAS-Preprint 2520.

See Also

`readMPMData`, `estimateESTATICS`

Examples

```
dataDir <- system.file("extdata",package="qMRI")
#
#  set file names for T1w, MTw and PDw images
#
t1Names <- paste0("t1w_",1:8,".nii.gz")
mtNames <- paste0("mtw_",1:6,".nii.gz")
pdNames <- paste0("pdw_",1:8,".nii.gz")
t1Files <- file.path(dataDir, t1Names)
mtFiles <- file.path(dataDir, mtNames)
pdFiles <- file.path(dataDir, pdNames)
#
#  file names of mask and B1 field map
#
B1File <- file.path(dataDir, "B1map.nii.gz")
maskFile <- file.path(dataDir, "mask.nii.gz")
#
#  Acquisition parameters (TE, TR, Flip Angle) for T1w, MTw and PDw images
#
TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
TR <- rep(25, 22)
FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
#
#   read MPM example data
#
library(qMRI)
mpm <- readMPMData(t1Files, pdFiles, mtFiles,
                   maskFile, TR = TR, TE = TE,
                   FA = FA, verbose = FALSE)
#
#  Estimate Parameters in the ESTATICS model
#
modelMPM <- estimateESTATICS(mpm, method = "NLR")
#
#  smooth maps of ESTATICS Parameters
#
setCores(2)
modelMPMQLsp1 <- smoothESTATICS(modelMPM,
                                kstar = 16,
                                alpha = 0.004,
                                patchsize=1,
                                verbose = TRUE)
#
# limit calculations to voxel in the central coronal slice
# to reduce execution time of the example
#
mpm$mask[,c(1:10,12:21),] <- FALSE
#
# Alternatively using Quasi-Likelihood
```

```
#
sigma <- 50
modelMPMQL <- estimateESTATICS(mpm, method = "QL",
                  sigma = array(sigma,mpm$sdim), L = 1)

#
#  resulting ESTATICS parameter maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
oldpar <- par(mfrow=c(2,4),mar=c(3,3,3,1),mgp=c(2,1,0))
pnames <- c("T1","MT","PD","R2star")
for(i in 1:4){
   rimage(modelMPMQL$modelCoeff[i,,11,])
   title(pnames[i])
   }
for(i in 1:4){
   rimage(modelMPMQLsp1$modelCoeff[i,,11,])
   title(paste("smoothed",pnames[i]))
   }
}
par(oldpar)
```

`writeESTATICS` *Write maps of ESTATICS parameters in standardized form as NIfTI files.*

Description

`R2`, `ST1`, `SPD` and, if available, `SMT`-maps are written as compressed NIfTI files into directory the speecified directory. If `class(mpmESTATICSModel) == "sESTATICSModel"` and an smoothed data are stored in `mpmESTATICSModel$smoothedData` the smoothed data are stored as ompressed NIfTI files in `dir` with filenames assembled using `prefix` and the names of the data source files.

Usage

```
writeESTATICS(mpmESTATICSModel, dir = NULL, prefix = "estatics", verbose = TRUE)
```

Arguments

`mpmESTATICSModel`	Object of class 'ESTATICSModel' or 'sESTATICSModel' as returned from function `estimateESTATICS` or `smoothESTATICS`.
`dir`	Directory name (or path) for output.
`prefix`	Prefix for file names
`verbose`	logical - provide information on progress

Value

The function returns NULL

Author(s)

Karsten Tabelow <tabelow@wias-berlin.de>
J\"org Polzehl <polzehl@wias-berlin.de>

See Also

readMPMData, estimateESTATICS, smoothESTATICS

Examples

```
dataDir <- system.file("extdata",package="qMRI")
outDir <- tempdir()
#
#  set file names for T1w, MTw and PDw images
#
t1Names <- paste0("t1w_",1:8,".nii.gz")
mtNames <- paste0("mtw_",1:6,".nii.gz")
pdNames <- paste0("pdw_",1:8,".nii.gz")
t1Files <- file.path(dataDir, t1Names)
mtFiles <- file.path(dataDir, mtNames)
pdFiles <- file.path(dataDir, pdNames)
#
#  file names of mask and B1 field map
#
B1File <- file.path(dataDir, "B1map.nii.gz")
maskFile <- file.path(dataDir, "mask.nii.gz")
#
#  Acquisition parameters (TE, TR, Flip Angle) for T1w, MTw and PDw images
#
TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
TR <- rep(25, 22)
FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
#
#   read MPM example data
#
library(qMRI)
mpm <- readMPMData(t1Files, pdFiles, mtFiles,
                   maskFile, TR = TR, TE = TE,
                   FA = FA, verbose = FALSE)
#
# limit calculations to voxel in the central coronal slice
# to reduce execution time of the example
#
mpm$mask[,c(1:10,12:21),] <- FALSE
#
#  Estimate Parameters in the ESTATICS model
```

```
#
modelMPM <- estimateESTATICS(mpm, method = "NLR")
#
#  resulting ESTATICS parameter maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
oldpar <- par(mfrow=c(2,2),mar=c(3,3,3,1),mgp=c(2,1,0))
pnames <- c("T1","MT","PD","R2star")
for(i in 1:4){
   rimage(modelMPM$modelCoeff[i,,11,])
   title(pnames[i])
   }
}
#
#  write ESTATICS parameter maps
#
writeESTATICS(modelMPM, dir=outDir, prefix="estatics")
par(oldpar)
```

`writeQI` *Write estimated maps in standardized form as NIfTI files.*

Description

Quantitative `R2`, `R1`, `PD` and, if available, `MT`-maps are written as compressed NIfTI files into directory the specified directory.

Usage

```
writeQI(qi, dir = NULL, prefix="qmap", verbose = TRUE)
```

Arguments

`qi`	Object of class 'qMaps' as returned from function `calculateQI`
`dir`	Directory name (or path) for output.
`prefix`	Prefix for file names
`verbose`	logical - provide information on progress

Value

The function returns `NULL`

Author(s)

Karsten Tabelow <tabelow@wias-berlin.de>
J\"org Polzehl <polzehl@wias-berlin.de>

See Also

readMPMData, estimateESTATICS,calculateQI

Examples

```
dataDir <- system.file("extdata",package="qMRI")
outDir <- tempdir()
#
#  set file names for T1w, MTw and PDw images
#
t1Names <- paste0("t1w_",1:8,".nii.gz")
mtNames <- paste0("mtw_",1:6,".nii.gz")
pdNames <- paste0("pdw_",1:8,".nii.gz")
t1Files <- file.path(dataDir, t1Names)
mtFiles <- file.path(dataDir, mtNames)
pdFiles <- file.path(dataDir, pdNames)
#
#  file names of mask and B1 field map
#
B1File <- file.path(dataDir, "B1map.nii.gz")
maskFile <- file.path(dataDir, "mask.nii.gz")
#
#  Acquisition parameters (TE, TR, Flip Angle) for T1w, MTw and PDw images
#
TE <- c(2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8,
        2.3, 4.6, 6.9, 9.2, 11.5, 13.8, 16.1, 18.4)
TR <- rep(25, 22)
FA <- c(rep(21, 8), rep(6, 6), rep(6, 8))
#
#   read MPM example data
#
library(qMRI)
mpm <- readMPMData(t1Files, pdFiles, mtFiles,
                   maskFile, TR = TR, TE = TE,
                   FA = FA, verbose = FALSE)
#
# limit calculations to voxel in the central coronal slice
# to reduce execution time of the example
#
mpm$mask[,c(1:10,12:21),] <- FALSE
#
#  Estimate Parameters in the ESTATICS model
#
modelMPM <- estimateESTATICS(mpm, method = "NLR")
#
#  resulting ESTATICS parameter maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
oldpar <- par(mfrow=c(2,2),mar=c(3,3,3,1),mgp=c(2,1,0))
pnames <- c("T1","MT","PD","R2star")
```

```
for(i in 1:4){
   rimage(modelMPM$modelCoeff[i,,11,])
   title(pnames[i])
   }
}
#
#  Compute quantitative maps (R1, R2star, PD, MT)
#
qMRIMaps <- calculateQI(modelMPM,
                        b1File = B1File,
                        TR2 = 3.4)
#
#  resulting quantitative maps for central coronal slice
#
if(require(adimpro)){
rimage.options(zquantiles=c(.01,.99), ylab="z")
par(mfrow=c(2,2),mar=c(3,3,3,1),mgp=c(2,1,0))
nmaps <- c("R1","R2star","PD","MT")
qmap <- extract(qMRIMaps,nmaps)
for (i in 1:4) rimage(qmap[[i]][,11,],main=nmaps[i])
}
#
#  write qmaps
#
writeQI(qMRIMaps, dir=outDir, prefix="qmap")
par(oldpar)
```

Index