Package ‘qMRI’

July 22, 2025

Type Package

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

Version 1.2.7.9

Date 2025-03-05

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

Depends R (>= 3.5), awsMethods (>= 1.0), methods, parallel

Imports oro.nifti (>= 0.9), stringr, aws (>= 2.4), 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>) and analysis of Inversion Recovery MRI data.
Usage of the package is described in
Polzehl and Tabelow (2023),
``Magnetic Resonance Brain Imaging'', 2nd Edition, Chapter 6 and 7, Springer, Use R! Series.
<doi:10.1007/978-3-031-38949-8>.
J. Polzehl and K. Tabelow (2023), ``Magnetic Resonance Brain Imaging -
Modeling and Data Analysis Using R: Code and Data.''
<doi:10.20347/WIAS.DATA.6> provides extensive example code and data.

URL `https://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 2025-03-06 14:20:05 UTC

Contents

`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>) and analysis of Inversion Recovery MRI data. Usage of the package is described in Polzehl and Tabelow (2023), "Magnetic Resonance Brain Imaging", 2nd Edition, Chapter 6 and 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8>. J. Polzehl and K. Tabelow (2023), "Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data." <doi:10.20347/WIAS.DATA.6> provides extensive example code and data.

Details

The DESCRIPTION file:

Package:	qMRI
Type:	Package
Title:	Methods for Quantitative Magnetic Resonance Imaging (’qMRI’)
Version:	1.2.7.9
Date:	2025-03-05
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 (>= 3.5), awsMethods (>= 1.0), methods, parallel

Imports:	oro.nifti (>= 0.9), stringr, aws (>= 2.4), adimpro (>= 0.9)
LazyData:	TRUE
Description:	Implementation of methods for estimation of quantitative maps from Multi-Parameter Mapping (MPM)
License:	GPL (>= 2)
Copyright:	This package is Copyright (C) 2015-2024 Weierstrass Institute for Applied Analysis and Stochastics.
URL:	https://www.wias-berlin.de/research/ats/imaging/
Suggests:	covr, testthat, knitr, rmarkdown
VignetteBuilder:	knitr
RoxygenNote:	6.1.1
NeedsCompilation:	yes
Packaged:	2025-03-05 12:33:41 UTC; tabelow
Author:	Joerg Polzehl [aut], Karsten Tabelow [aut, cre], WIAS Berlin [cph, fnd]

Index of help topics:

```
MREdisplacement         Calculate the motion induced signal phase for
                        IR-MRE in biphasic material
awssigmc                Estimate noise variance for multicoil MR
                        systems
calculateQI             Obtain quantitative maps from estimated
                        ESTATICS parameters.
colMT                   MT map color scheme
estimateESTATICS        Estimate parameters in the ESTATICS model.
estimateIR              Estimate IRMRI parameters
estimateIRfluid         Estimate parameters in Inversion Recovery MRI
                        experiments model for CSF voxel
estimateIRsolid         Estimate parameters in Inversion Recovery MRI
                        experiments mixture model for non-fluid voxel
estimateIRsolidfixed    Estimate mixture parameter in Inversion
                        Recovery MRI experiments mixture model for
                        non-fluid voxel
extract.ANY-method      Methods to extract information from objects of
                        class '”MPMData”', '”ESTATICSModel”',
                        '”sESTATICSModel”', '”qMaps”', '”IRdata”',
                        '”IRfluid”' and '”IRmixed”'.
generateIRData          generate IR MRI example data
qMRI-package            Methods for Quantitative Magnetic Resonance
                        Imaging ('qMRI')
readIRData              Prepare IRMRI dataset
readMPMData             Read experimental Multi-Parameter Mapping (MPM)
                        data.
smoothESTATICS          Adaptive smoothing of ESTATICS parameters and
                        MPM data
smoothIRSolid           Smooth object generated by function
                        'estimateIRsolid'
writeESTATICS           Write maps of ESTATICS parameters in
                        standardized form as NIfTI files.
```

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

Further information is available in the following vignettes:

`IRMRI-Example`	An example session for analyzing Inversion Recovery MRI and MR Elastography data (source, pdf)
`qMRI-Example`	An example session for analyzing quantitative MRI data in the Multi-Parameter Mapping framework (sourc

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

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 6 and 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

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)
modelMPMsp1 <- smoothESTATICS(modelMPM,
                              kstar = 16,
                              alpha = 0.004,
                              patchsize=1,
                              verbose = TRUE)
#
#  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))
on.exit(par(oldpar))
pnames <- c("T1","MT","PD","R2star")
modelCoeff <- extract(modelMPM,"modelCoeff")
for(i in 1:4){
   rimage(modelCoeff[i,,11,])
   title(pnames[i])
   }
   modelCoeff <- extract(modelMPMsp1,"modelCoeff")
for(i in 1:4){
   rimage(modelCoeff[i,,11,])
   title(paste("smoothed",pnames[i]))
   }
}
#
#  Compute quantitative maps (R1, R2star, PD, MT)
#
qMRIMaps <- calculateQI(modelMPM,
                        b1File = B1File,
                        TR2 = 3.4)
```

```
qMRISmoothedp1Maps <- calculateQI(modelMPMsp1,
                                  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])
qmap <- extract(qMRISmoothedp1Maps,nmaps)
for (i in 1:4) rimage(qmap[[i]][,11,],main=paste("Smoothed",nmaps[i]))
}
par(oldpar)
```

`awssigmc` *Estimate noise variance for multicoil MR systems*

Description

The distribution of image intensity values S_i divided by the noise standard deviation in K-space σ in dMRI experiments is assumed to follow a non-central chi-distribution with $2L$ degrees of freedom and noncentrality parameter η, where L refers to the number of receiver coils in the system and $\sigma\eta$ is the signal of interest. This is an idealization in the sense that each coil is assumed to have the same contribution at each location. For realistic modeling L should be a locally smooth function in voxel space that reflects the varying local influence of the receiver coils in the the reconstruction algorithm used.

The functions assume L to be known and estimate either a local (function `awslsigmc`) or global (function `awssigmc`) σ employing an assumption of local homogeneity for the noncentrality parameter η.

Function `afsigmc` implements estimates from Aja-Fernandez (2009). Function `aflsigmc` implements the estimate from Aja-Fernandez (2013).

Usage

```
awssigmc(y, steps, mask = NULL, ncoils = 1, vext = c(1, 1), lambda = 20,
         h0 = 2, verbose = FALSE, sequence = FALSE, hadj = 1, q = 0.25,
         qni = .8, method=c("VAR","MAD"))
awslsigmc(y, steps, mask = NULL, ncoils = 1, vext = c(1, 1), lambda = 5, minni = 2,
         hsig = 5, sigma = NULL, family = c("NCchi"), verbose = FALSE,
         trace=FALSE, u=NULL)
```

Arguments

`y`	3D array, usually obtained from an object of class `dwi` as `obj@si[,,,i]` for some `i`, i.e. one 3D image from an dMRI experiment. Alternatively a vector of length `sum(mask)` may be suppied together with a brain mask in `mask`.
`steps`	number of steps in adapive weights smoothing, used to reveal the unerlying mean structure.
`mask`	restrict computations to voxel in mask, if `is.null(mask)` all voxel are used. In function `afsigmc` mask should refer to background for `method %in% c("modem1chi","bkm2chi","bkm1` and to voxel within the head for `method=="modevn"`.
`ncoils`	number of coils, or equivalently number of effective degrees of freedom of non-central chi distribution divided by 2.
`vext`	voxel extentions
`lambda`	scale parameter in adaptive weights smoothing
`h0`	initial bandwidth
`verbose`	if `verbose==TRUE` density plots and quantiles of local estimates of `sigma` are provided.
`trace`	if `trace==TRUE` intermediate results for each step are returned in component tergs for all voxel in mask.
`sequence`	if `sequence=TRUE` a vector of estimates for the noise standard deviation `sigma` for the individual steps is returned instead of the final value only.
`hadj`	adjustment factor for bandwidth (chosen by `bw.nrd`) in mode estimation
`q`	quantile to be used for interquantile-differences.
`qni`	quantile of distribution of actual sum of weights $N_i = \sum_j w_{ij}$ in adaptive smoothing. Only voxel i with $N_i > q_{qni}(N_.)$ are used for variance estimation. Should be larger than 0.5.
`method`	in case of function `awssigmc` the method for variance estimation, either "VAR" (variance) or "MAD" (mean absolute deviation). In function `afsigmc` see last column in Table 2 in Aja-Fernandez (2009).
`minni`	Minimum sum of weights for updating values of `sigma`.
`hsig`	Bandwidth of the median filter.
`sigma`	Initial estimate for `sigma`
`family`	One of `"Gauss"` or `"NCchi"` (default) defining the probability distribution to use.
`u`	if `verbose==TRUE` an array of noncentrality paramters for comparisons. Internal use for tests only

Value

a list with components

`sigma`	either a scalar or a vector of estimated noise standard deviations.
`theta`	the estimated mean structure

Author(s)

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

References

K. Tabelow, H.U. Voss, J. Polzehl, Local estimation of the noise level in MRI using structural adaptation, Medical Image Analysis, 20 (2015), pp. 76–86.

See Also

`aws`

`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

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 6, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_6>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

`readMPMData`, `estimateESTATICS`, `smoothESTATICS`, `writeESTATICS`, `awsLocalSigma`

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, "mask0.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
#
#
#  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))
on.exit(par(oldpar))
pnames <- c("T1","MT","PD","R2star")
modelCoeff <- extract(modelMPM,"modelCoeff")
for(i in 1:4){
   rimage(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,
                 varest = c("RSS", "data"), 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.

`varest`	For parameter covariance estimation use either residual sum of squares (RSS) or estimate variances for T1, MT (is available) and PD from higest intensity images using function `awsLocalSigma`from package **aws**.
`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

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 6, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_6>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

`readMPMData`, `calculateQI`, `smoothESTATICS`, `writeESTATICS`, `awsLocalSigma`

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, "mask0.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")
# Alternatively using Quasi-Likelihood
```

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

`estimateIR` *Estimate IRMRI parameters*

Description

Parameter estimation (intensity, relaxation rate, proportion of fluid) in Inversion Recovery MRI data.

Usage

```
estimateIR(IRdataobj, TEScale = 100, dataScale = 1000, method = c("NLR", "QL"),
         varest = c("RSS","data"), fixed = TRUE, smoothMethod=c("PAWS","Depth"),
           kstar = 24, alpha = .025, bysegment = TRUE, verbose = TRUE)
```

Arguments

`IRdataobj`	4D array of IRMRI signals. First dimension corresponds to Inversion times (InvTime).
`TEScale`	Internal scale factor for Echo Times. This influences parameter scales in numerical calculations.
`dataScale`	Internal scale factor for MR signals. This influences parameter scales in numerical calculations.
`method`	Either `"NLS"` for nonlinear least squares (ignores Rician bias) or `"QL"` for Quasi-Likelihood. The second option is more accurate but requires additional information and is computationally more expensive.
`varest`	Method to, in case of `method="QR"`, estimate `sigma`if not provided. Either from residual sums of squares (`"RSS"`) or MR signals (`"data"`) using function `varest` from package aws. Only to be used in case that no image registration was needed as preprocessing.
`fixed`	Should adaptive smoothing performed for Sx and Rx maps and fx maps reestimated afterwards ?
`smoothMethod`	Either "PAWS" or "Depth". the second option is not yet implemented.
`kstar`	number of steps used in PAWS
`alpha`	significance level for decisions in aws algorithm (suggestion: between 1e-5 and 0.025)
`bysegment`	`TRUE`: restrict smoothing to segments from segmentation, `FALSE`: restrict smoothing to solid mask.
`verbose`	Logical. Provide some runtime diagnostics.

Details

This function implements the complete pipeline of IRMRI anlysis.

Value

List of class `"IRmixed"` with components

`IRdata`	4D array containing the IRMRI data, first dimension refers to inversion times
`InvTimes`	vector of inversion times
`segm`	segmentation codes, 1 for CSF, 2 for GM, 3 for WM, 0 for out of brain
`sigma`	noise standard deviation, if not specified estimated fron CSF areas in image with largest inversion time
`L`	effective number of coils
`fx`	Array of fluid proportions
`Sx`	Array of maximal signals
`Rx`	Array of relaxation rates
`Sf`	Global estimate of maximal fluid signal
`Rf`	Global estimate of fluid relaxation rate
`ICovx`	Covariance matrix of estimates `fx`, `Sx` and `Rx`.
`sigma`	Array of provided or estimated noise standard deviations
`Convx`	Array of convergence indicators
`rsdx`	Residual standard deviations

The arrays contain entries for all voxel with `segments%in%1:3`.

Author(s)

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

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_7>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

`estimateIRfluid`, `estimateIRsolid`, `estimateIRsolidfixed`,`smoothIRSolid`

Examples

```
## runs about 30 seconds
dataDir0 <- system.file("extdataIR", package = "qMRI")
dataDir <- tempdir()
library(oro.nifti)
library(qMRI)
segm <- readNIfTI(file.path(dataDir0,"Brainweb_segm"))
Sf <- 900
Rf <- 0.000285
Sgm <- 400
Rgm <- 0.00075
fgm <- .15
Swm <- 370
Rwm <- 0.0011
fwm <- .05
InvTimes0 <- c(100, 200, 400, 600, 800, 1200, 1600, 2000, 2500, 3000,
              3500, 4000, 4500, 5000, 6000, 15000)
nTimes <- length(InvTimes0)
sigma <- 40
## generate IR signal
IRdata <- generateIRData(segm, c(Sf,Rf), c(fgm,Rgm,Sgm), c(fwm,Rwm,Swm), InvTimes0, sigma)
for(i in 1:9) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR0",i)))
for(i in 10:nTimes) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR",i)))
## generate IRdata object
t1Files <- list.files(dataDir,"*.nii.gz",full.names=TRUE)
segmFile <- file.path(dataDir0,"Brainweb_segm")
IRdata <- readIRData(t1Files, InvTimes0, segmFile, sigma=sigma,
                     L=1, segmCodes=c("CSF","GM","WM"))
## estimate all
sIRmix <- estimateIR(IRdata, method="QL")
```

`estimateIRfluid` *Estimate parameters in Inversion Recovery MRI experiments model for CSF voxel*

Description

The Inversion Recovery MRI signal in voxel containing only CSF follows is modeled as $S_InvTime = par[1] * abs(1 - 2 * exp(-InvTime*par[2]))$ dependings on two parameters. These parameters are assumed to be tissue (and scanner) dependent.

Usage

```
estimateIRfluid(IRdataobj, TEScale = 100, dataScale = 1000,
method = c("NLR", "QL"), varest = c("RSS", "data"),
verbose = TRUE, lower = c(0, 0), upper = c(2, 2))
```

Arguments

`IRdataobj`	Object of class `"IRdata"` as generated by function `readIRData`.
`TEScale`	Internal scale factor for Echo Times. This influences parameter scales in numerical calculations.
`dataScale`	Internal scale factor for MR signals. This influences parameter scales in numerical calculations.
`method`	Either `"NLS"` for nonlinear least squares (ignores Rician bias) or `"QL"` for Quasi-Likelihood. The second option is more accurate but requires additional information and is computationally more expensive.
`varest`	Method to, in case of `method="QR"`, estimate `sigma`if not provided. Either from residual sums of squares (`"RSS"`) or MR signals (`"data"`) using function `varest` from package `aws`. Only to be used in case that no image registration was needed as preprocessing.
`verbose`	Logical. Provide some runtime diagnostics.
`lower`	Lower bounds for parameter values.
`upper`	Upper bounds for parameter values.

Details

The Inversion Recovery MRI signal in voxel containing only CSF follows is modeled as $S_InvTime = par[1] * abs(1 - 2 * exp(-InvTime*par[2]))$ dependings on two parameters. These parameters are assumed to be tissue (and scanner) dependent.

Value

List of class `IRfluid` with components

`IRdata`	4D array containing the IRMRI data, first dimension refers to inversion times
`InvTimes`	vector of inversion times
`segm`	segmentation codes, 1 for CSF, 2 for GM, 3 for WM, 0 for out of brain
`sigma`	noise standard deviation, if not specified estimated fron CSF areas in image with largest inversion time
`L`	effective number of coils
`Sf`	Global estimate of maximal fluid signal
`Rf`	Global estimate of fluid relaxation rate
`Sx`	Array of maximal signals
`Rx`	Array of relaxation rates
`sigma`	Array of provided or estimated noise standard deviations
`Convx`	Array of convergence indicators
`method`	`"NLS"` for nonlinear regression or `"QL"` for quasi likelihood.
`varest`	Method used for variance estimation

The arrays only contain entries for fluid voxel.

Author(s)

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

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_7>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

estimateIR, estimateIRsolid, estimateIRsolidfixed,smoothIRSolid

Examples

```
dataDir0 <- system.file("extdataIR", package = "qMRI")
dataDir <- tempdir()
library(oro.nifti)
library(qMRI)
segm <- readNIfTI(file.path(dataDir0,"Brainweb_segm"))
Sf <- 900
Rf <- 0.000285
Sgm <- 400
Rgm <- 0.00075
fgm <- .15
Swm <- 370
Rwm <- 0.0011
fwm <- .05
InvTimes0 <- c(100, 200, 400, 600, 800, 1200, 1600, 2000, 2500, 3000,
              3500, 4000, 4500, 5000, 6000, 15000)
nTimes <- length(InvTimes0)
sigma <- 40
## generate IR signal
IRdata <- generateIRData(segm, c(Sf,Rf), c(fgm,Rgm,Sgm), c(fwm,Rwm,Swm), InvTimes0, sigma)
for(i in 1:9) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR0",i)))
for(i in 10:nTimes) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR",i)))
## generate IRdata object
t1Files <- list.files(dataDir,"*.nii.gz",full.names=TRUE)
segmFile <- file.path(dataDir0,"Brainweb_segm")
IRdata <- readIRData(t1Files, InvTimes0, segmFile, sigma=sigma,
                     L=1, segmCodes=c("CSF","GM","WM"))
## estimate fluid
setCores(2) # parallel mode using 2 threads
IRfluid <- estimateIRfluid(IRdata, method="NLR", verbose=FALSE)
cat("Estimated parameters Sf:", IRfluid$Sf,
                        " Rf:", IRfluid$Rf, "\n")
```

`estimateIRsolid` *Estimate parameters in Inversion Recovery MRI experiments mixture model for non-fluid voxel*

Description

The Inversion Recovery MRI signal in non-fluid voxel follows is modeled as a mixture of a fluid and a solid compartment.

Usage

```
estimateIRsolid(IRfluidobj, TEScale = 100, dataScale = 1000,
verbose = TRUE, lower = c(0, 0, 0), upper = c(0.95, 2, 2))
```

Arguments

`IRfluidobj`	Object of class `"IRfluid"` as generated by function `estimateIRfluid`.
`TEScale`	Internal scale factor for Echo Times. This influences parameter scales in numerical calculations.
`dataScale`	Internal scale factor for MR signals. This influences parameter scales in numerical calculations.
`verbose`	Logical. Provide some runtime diagnostics.
`lower`	Lower bounds for parameter values.
`upper`	Upper bounds for parameter values.

Details

The Inversion Recovery MRI signal in non-fluid voxel follows is modeled as a mixture of a fluid and a solid compartment. The function calculates estimates of the maximum signal and recovery rate for the solid compartment and a mixture coefficient (proportion of fluid) for all voxel with `segment%in%2:3`using results from function `estimateIRfluid`.

Value

List of class `IRmixed` with components

`IRdata`	4D array containing the IRMRI data, first dimension refers to inversion times
`InvTimes`	vector of inversion times
`segm`	segmentation codes, 1 for CSF, 2 for GM, 3 for WM, 0 for out of brain
`sigma`	noise standard deviation, if not specified estimated fron CSF areas in image with largest inversion time
`L`	effective number of coils
`fx`	Array of fluid proportions
`Sx`	Array of maximal signals

`Rx`	Array of relaxation rates
`Sf`	Global estimate of maximal fluid signal
`Rf`	Global estimate of fluid relaxation rate
`ICovx`	Covariance matrix of estimates `fx`, `Sx` and `Rx`.
`sigma`	Array of provided or estimated noise standard deviations
`Convx`	Array of convergence indicators
`rsdx`	Residual standard deviations
`method`	`"NLS"` for nonlinear regression or `"QL"` for quasi likelihood.
`varest`	Method used for variance estimation

The arrays contain entries for all voxel with `segments%in%1:3`.

Author(s)

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

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_7>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

`estimateIRfluid`, `estimateIR`, `estimateIRsolidfixed`,`smoothIRSolid`

Examples

```
## runs about 7 seconds
dataDir0 <- system.file("extdataIR", package = "qMRI")
dataDir <- tempdir()
library(oro.nifti)
library(qMRI)
segm <- readNIfTI(file.path(dataDir0,"Brainweb_segm"))
Sf <- 900
Rf <- 0.000285
Sgm <- 400
Rgm <- 0.00075
fgm <- .15
Swm <- 370
Rwm <- 0.0011
fwm <- .05
InvTimes0 <- c(100, 200, 400, 600, 800, 1200, 1600, 2000, 2500, 3000,
              3500, 4000, 4500, 5000, 6000, 15000)
nTimes <- length(InvTimes0)
sigma <- 40
```

```
## generate IR signal
IRdata <- generateIRData(segm, c(Sf,Rf), c(fgm,Rgm,Sgm), c(fwm,Rwm,Swm), InvTimes0, sigma)
for(i in 1:9) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR0",i)))
for(i in 10:nTimes) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR",i)))
## generate IRdata object
t1Files <- list.files(dataDir,"*.nii.gz",full.names=TRUE)
segmFile <- file.path(dataDir0,"Brainweb_segm")
IRdata <- readIRData(t1Files, InvTimes0, segmFile, sigma=sigma,
                     L=1, segmCodes=c("CSF","GM","WM"))
## estimate fluid
setCores(2) # parallel mode using 2 threads
IRfluid <- estimateIRfluid(IRdata, method="NLR", verbose=FALSE)
cat("Estimated parameters Sf:", IRfluid$Sf,
                        " Rf:", IRfluid$Rf, "\n")
## estimate solid
IRmix <- estimateIRsolid(IRfluid, verbose=FALSE)
```

`estimateIRsolidfixed` *Estimate mixture parameter in Inversion Recovery MRI experiments mixture model for non-fluid voxel*

Description

Reestimate proportion of fluid with Sx and Rx fixed after smoothing.

Usage

```
estimateIRsolidfixed(IRmixedobj, TEScale = 100, dataScale = 1000,
verbose = TRUE, lower = c(0), upper = c(0.95))
```

Arguments

`IRmixedobj`	Object of class `"IRmixed"` as generated by function `smoothIRSolid` or `estimateIRsolid`.
`TEScale`	Internal scale factor for Echo Times. This influences parameter scales in numerical calculations.
`dataScale`	Internal scale factor for MR signals. This influences parameter scales in numerical calculations.
`verbose`	Logical. Provide some runtime diagnostics.
`lower`	lower bound for fx (fluid proportion)
`upper`	upper bound for fx (fluid proportion)

Value

List of class `"IRmixed"` components

`IRdata`	4D array containing the IRMRI data, first dimension refers to inversion times
`InvTimes`	vector of inversion times
`segm`	segmentation codes, 1 for CSF, 2 for GM, 3 for WM, 0 for out of brain
`sigma`	noise standard deviation, if not specified estimated fron CSF areas in image with largest inversion time
`L`	effective number of coils
`fx`	Array of fluid proportions
`Sx`	Array of maximal signals
`Rx`	Array of relaxation rates
`Sf`	Global estimate of maximal fluid signal
`Rf`	Global estimate of fluid relaxation rate
`ICovx`	Covariance matrix of estimates `fx`, `Sx` and `Rx`.
`sigma`	Array of provided or estimated noise standard deviations
`Convx`	Array of convergence indicators
`rsdx`	Residual standard deviations
`method`	`"NLS"` for nonlinear regression or `"QL"` for quasi likelihood.
`varest`	Method used for variance estimation

The arrays contain entries for all voxel with `segments%in%1:3`.

Author(s)

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

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_7>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

`estimateIRfluid`, `estimateIRsolid`, `estimateIR`,`smoothIRSolid`, `readIRData`

Examples

```
## runs about 11 seconds
dataDir0 <- system.file("extdataIR", package = "qMRI")
dataDir <- tempdir()
library(oro.nifti)
library(qMRI)
segm <- readNIfTI(file.path(dataDir0,"Brainweb_segm"))
Sf <- 900
Rf <- 0.000285
Sgm <- 400
Rgm <- 0.00075
fgm <- .15
Swm <- 370
Rwm <- 0.0011
fwm <- .05
InvTimes0 <- c(100, 200, 400, 600, 800, 1200, 1600, 2000, 2500, 3000,
              3500, 4000, 4500, 5000, 6000, 15000)
nTimes <- length(InvTimes0)
sigma <- 40
## generate IR signal
IRdata <- generateIRData(segm, c(Sf,Rf), c(fgm,Rgm,Sgm), c(fwm,Rwm,Swm), InvTimes0, sigma)
for(i in 1:9) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR0",i)))
for(i in 10:nTimes) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR",i)))
## generate IRdata object
t1Files <- list.files(dataDir,"*.nii.gz",full.names=TRUE)
segmFile <- file.path(dataDir0,"Brainweb_segm")
IRdata <- readIRData(t1Files, InvTimes0, segmFile, sigma=sigma,
                     L=1, segmCodes=c("CSF","GM","WM"))
## estimate fluid
setCores(2) # parallel mode using 2 threads
IRfluid <- estimateIRfluid(IRdata, method="NLR", verbose=FALSE)
cat("Estimated parameters Sf:", IRfluid$Sf,
                        " Rf:", IRfluid$Rf, "\n")
## estimate solid
IRmix <- estimateIRsolid(IRfluid, verbose=FALSE)
## smoothing
sIRmix <- smoothIRSolid(IRmix, alpha=1e-4, partial=FALSE, verbose=FALSE)
## reestimate
sIRmix <- estimateIRsolidfixed(sIRmix, verbose=FALSE)
```

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

Description

The extract-methods extract and/or compute specified statistics from object of class `"MPMData"`, `"ESTATICSModel"`, `"sESTATICSModel"`, `"qMaps"`, `"IRdata"`, `"IRfluid"` and `"IRmixed"`. The [-methods can be used to reduce objects of class `"MPMData"`, `"ESTATICSModel"`, `"sESTATICSModel"`, `"qMaps"`, `"IRdata"`, `"IRfluid"` and `"IRmixed"` such that they contain a subcube of data and results.

Usage

```
## S3 method for class 'MPMData'
extract(x, what, ...)
## S3 method for class 'ESTATICSModel'
extract(x, what, ...)
## S3 method for class 'sESTATICSModel'
extract(x, what, ...)
## S3 method for class 'qMaps'
extract(x, what, ...)
## S3 method for class 'IRdata'
extract(x, what, ...)
## S3 method for class 'IRfluid'
extract(x, what, ...)
## S3 method for class 'IRmixed'
extract(x, what, ...)
## S3 method for class 'MPMData'
x[i, j, k, ...]
## S3 method for class 'ESTATICSModel'
x[i, j, k, ...]
## S3 method for class 'sESTATICSModel'
x[i, j, k, ...]
## S3 method for class 'qMaps'
x[i, j, k, ...]
## S3 method for class 'IRdata'
x[i, j, k, tind, ...]
## S3 method for class 'IRfluid'
x[i, j, k, ...]
## S3 method for class 'IRmixed'
x[i, j, k, ...]
```

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.
`i`	index vector for first spatial dimension
`j`	index vector for second spatial dimension
`k`	index vector for third spatial dimension
`tind`	index vector for inversion times

...	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: standard deviation sigma
- L: effective number of receiver coils 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: effective number of receiver coils 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`>

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_7>.

`generateIRData` *generate IR MRI example data*

Description

The function generates IR MRI example data for specified parameters

Usage

```
generateIRData(segm, pCSF, pGM, pWM, InvTimes, sigma = 40)
```

Arguments

`segm`	array containing segmentation resuts for an 3D MRI template. Contains 1 for CSF, 2 for Gray Matter and 3 for White Matter
`pCSF`	Parameters (S,R) for CSF
`pGM`	Parameters (f,R,S) for Gray Matter
`pWM`	Parameters (f,R,S) for White Matter
`InvTimes`	Vector of Inversion Times, length nTimes
`sigma`	Noise standard variation

Value

array with dimension c(nTimes,dim(segm))

Note

used in examples for IR functions

Author(s)

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

See Also

`estimateIRfluid`, `estimateIRsolid`, `estimateIR`,`smoothIRSolid`, `readIRData`

MREdisplacement *Calculate the motion induced signal phase for IR-MRE in biphasic material*

Description

The function takes magnitude images and phase images (as NIfTI files) recordet with inversion `IT1=Inf` and a second inversion time `IT2` that nulls the fluid signal. Tissue parameters (Relaxation rates) are extracted from an object of class `"IRmixed"` calculated from data of a related IRMRI experiment.

Usage

```
MREdisplacement(MagnFiles1, PhaseFiles1, MagnFiles2, PhaseFiles2, TI2 = 2400,
             IRmixobj, method = c("full", "approx"),rescale=FALSE,verbose=FALSE)
```

Arguments

`MagnFiles1`	Filenames of magnitude images recorded with inversion time `IT=Inf` .
`PhaseFiles1`	Filenames of phase images recorded with inversion time `IT=Inf` .
`MagnFiles2`	Filenames of magnitude images recorded with inversion time `IT=IT2`.
`PhaseFiles2`	Filenames of phase images recorded with inversion time `IT=IT2` .
`TI2`	Inversion time used for `MagnFiles2` and `PhaseFiles2`. `IT2` should be selected to extinguish the signal intendity for fluid.
`IRmixobj`	Object of class `"IRmixed"` obtained from a related IRMRI experiment.
`method`	Either `"full"` or `"approx"`
`rescale`	Logical, do we need to rescale phase images ?
`verbose`	Report scale range of phase images

Details

The first 4 arguments need to be vectors of filenames of identical length with files containing compatible 3D NIfTI images. Object `IRmixobj` needs to contain a components `segm` and `Rx` of compatible dimension that need to be registered to the MRE images.

Value

A list of class `"IRMREbiphasic"` with components

`phisolid`	displacement solid
`phifluid`	displacement fluid

Author(s)

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

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_7>.

See Also

`estimateIRfluid`, `estimateIRsolid`, `estimateIRsolidfixed`,`smoothIRSolid`

`readIRData` *Prepare IRMRI dataset*

Description

The function reads IRMRI images given as NIfTI files in t1Files, inversion times and segmentation image(s) aund prepares an object class `"IRdata"`

Usage

```
readIRData(t1Files, InvTimes, segmFile, sigma = NULL, L = 1,
           segmCodes = c("GM", "WM", "CSF"))
```

Arguments

`t1Files`	Names of NIfTI files containing the recorded images.
`InvTimes`	Corresponding inversion times
`segmFile`	Either a NIfTI file containing a segmentation into GM, WM and CSF or three files containing probability maps for GM, WM and CSF
`sigma`	Noise standard deviation
`L`	Effective number of coils, L=1 assumes a Rician signal distribution
`segmCodes`	sequence of tissue code in segmFile

Value

A list of class `"IRdata"` with components

`IRdata`	4D array containing the IRMRI data, first dimension refers to inversion times
`InvTimes`	vector of inversion times
`segm`	segmentation codes, 1 for CSF, 2 for GM, 3 for WM, 0 for out of brain
`sigma`	noise standard deviation, if not specified estimated fron CSF areas in image with largest inversion time
`L`	effective number of coils

Author(s)

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

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_7>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

`estimateIRfluid`, `estimateIRsolid`, `estimateIR`,`smoothIRSolid`

Examples

```
dataDir0 <- system.file("extdataIR", package = "qMRI")
dataDir <- tempdir()
library(oro.nifti)
library(qMRI)
segm <- readNIfTI(file.path(dataDir0,"Brainweb_segm"))
Sf <- 900
Rf <- 0.000285
Sgm <- 400
Rgm <- 0.00075
fgm <- .15
Swm <- 370
Rwm <- 0.0011
fwm <- .05
InvTimes0 <- c(100, 200, 400, 600, 800, 1200, 1600, 2000, 2500, 3000,
              3500, 4000, 4500, 5000, 6000, 15000)
nTimes <- length(InvTimes0)
sigma <- 40
## generate IR signal
IRdata <- generateIRData(segm, c(Sf,Rf), c(fgm,Rgm,Sgm), c(fwm,Rwm,Swm), InvTimes0, sigma)
for(i in 1:9) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR0",i)))
for(i in 10:nTimes) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR",i)))
## generate IRdata object
t1Files <- list.files(dataDir,"*.nii.gz",full.names=TRUE)
segmFile <- file.path(dataDir0,"Brainweb_segm")
IRdata <- readIRData(t1Files, InvTimes0, segmFile, sigma=sigma,
                     L=1, segmCodes=c("CSF","GM","WM"))
```

`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, wghts = 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
`wghts`	optional weights for MPM data volumes. Only needed is volumes have different data variance, e.g., in case of averages of multiple acquisitions.
`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

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 6, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_6>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

`estimateESTATICS`, `calculateQI`, `smoothESTATICS`, `writeESTATICS`, `awsLocalSigma`

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, mscbw =5, 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.
`mscbw`	bandwidth for 3D median smoother used to stabilize the covariance estimates.
`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

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

J. Polzehl, K. Papafitsorus, K. Tabelow (2018). Patch-wise adaptive weights smoothing. WIAS-Preprint 2520.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 6, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_6>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

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)
modelMPMsp1 <- smoothESTATICS(modelMPM,
                              kstar = 16,
                              alpha = 0.004,
                              patchsize=1,
                              verbose = TRUE)
#
#  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))
on.exit(par(oldpar))
pnames <- c("T1","MT","PD","R2star")
modelCoeff <- extract(modelMPM,"modelCoeff")
for(i in 1:4){
   rimage(modelCoeff[i,,11,])
   title(pnames[i])
   }
modelCoeff <- extract(modelMPMsp1,"modelCoeff")
for(i in 1:4){
   rimage(modelCoeff[i,,11,])
   title(paste("smoothed",pnames[i]))
   }
}
par(oldpar)
```

---

smoothIRSolid *Smooth object generated by function* `estimateIRsolid`

Description

Adaptive smoothing of Rx and Sx maps over WM and GM areas.

Usage

```
smoothIRSolid(IRmixedobj, kstar = 24, patchsize = 1, alpha = 0.025,
                   mscbw = 5, bysegment=TRUE, partial=TRUE, verbose=TRUE)
```

Arguments

`IRmixedobj`	object of class `IRmixed` generated by function `estimateIRsolid`
`kstar`	number of steps for AWS algorithm
`patchsize`	patchsize in paws
`alpha`	significance level for decisions in aws algorithm (suggestion: between 1e-5 and 0.025)
`mscbw`	bandwidth for 3D median smoother used to stabilize the covariance estimates.
`bysegment`	`TRUE`: restrict smoothing to segments from segmentation, `FALSE`: restrict smoothing to solid mask.
`partial`	`TRUE`: ignore information concerning parameter `fx` when smoothing.
`verbose`	logical: Monitor process.

Details

This uses a vectorized version of the AWS algorithm that emloys inverse covariance estimates of the estimated parameters. Local smoothing is done for Rx and Sx maps in ergs which can be assumed to be locally smooth within tissue. No smoothing for fx maps since they may vary.

Value

an object of class `"IRmixed"`, but with components Sx and Rx replaced. The object carries an additional component `bi` containing an array of sum of weights characterizing the amount of smoothing.

Author(s)

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

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 7, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_7>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

See Also

estimateIRfluid, estimateIRsolid, estimateIRsolidfixed,estimateIR

Examples

```
## runs about 10 seconds
dataDir0 <- system.file("extdataIR", package = "qMRI")
dataDir <- tempdir()
library(oro.nifti)
library(qMRI)
segm <- readNIfTI(file.path(dataDir0,"Brainweb_segm"))
Sf <- 900
Rf <- 0.000285
Sgm <- 400
Rgm <- 0.00075
fgm <- .15
Swm <- 370
Rwm <- 0.0011
fwm <- .05
InvTimes0 <- c(100, 200, 400, 600, 800, 1200, 1600, 2000, 2500, 3000,
              3500, 4000, 4500, 5000, 6000, 15000)
nTimes <- length(InvTimes0)
sigma <- 40
## generate IR signal
IRdata <- generateIRData(segm, c(Sf,Rf), c(fgm,Rgm,Sgm), c(fwm,Rwm,Swm), InvTimes0, sigma)
for(i in 1:9) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR0",i)))
for(i in 10:nTimes) writeNIfTI(as.nifti(IRdata[i,,,]),
                         file.path(dataDir,paste0("IR",i)))
## generate IRdata object
t1Files <- list.files(dataDir,"*.nii.gz",full.names=TRUE)
segmFile <- file.path(dataDir0,"Brainweb_segm")
IRdata <- readIRData(t1Files, InvTimes0, segmFile, sigma=sigma,
                     L=1, segmCodes=c("CSF","GM","WM"))
## estimate fluid
setCores(2) # parallel mode using 2 threads
IRfluid <- estimateIRfluid(IRdata, method="NLR", verbose=FALSE)
cat("Estimated parameters Sf:", IRfluid$Sf,
                        " Rf:", IRfluid$Rf, "\n")
## estimate solid
IRmix <- estimateIRsolid(IRfluid, verbose=FALSE)
## smoothing
sIRmix <- smoothIRSolid(IRmix, alpha=1e-4, partial=FALSE, verbose=FALSE)
```

`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`>

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 6, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_6>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

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, "mask0.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")
#
#  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))
on.exit(par(oldpar))
pnames <- c("T1","MT","PD","R2star")
modelCoeff <- extract(modelMPM,"modelCoeff")
for(i in 1:4){
   rimage(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`>

References

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging: Modeling and Data Analysis Using R, 2nd Edition, Chapter 6, Springer, Use R! Series. <doi:10.1007/978-3-031-38949-8_6>.

J. Polzehl and K. Tabelow (2023), Magnetic Resonance Brain Imaging - Modeling and Data Analysis Using R: Code and Data. <doi:10.20347/WIAS.DATA.6>.

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, "mask0.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")
#
#  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))
on.exit(par(oldpar))
pnames <- c("T1","MT","PD","R2star")
modelCoeff <- extract(modelMPM,"modelCoeff")
for(i in 1:4){
   rimage(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