Package ‘dcemriS4’

April 29, 2015

Version  0.55
Date     2015-04-26
Title    A Package for Image Analysis of DCE-MRI (S4 Implementation)
Depends  R (>= 2.14.0), oro.nifti (>= 0.4.3)
Suggests bitops, minpack.lm, splines, XML, oro.dicom (>= 0.4.3),
          testthat
Imports  utils, parallel, methods
Description A collection of routines and documentation that allows one to
            perform voxel-wise quantitative analysis of dynamic contrast-enhanced MRI
            (DEC-MRI) and diffusion-weighted imaging (DWI) data, with emphasis on
            oncology applications.
License   BSD_3_clause + file LICENSE
LazyData  true
LazyDataCompression gzip
NeedsCompilation yes
Author    Brandon Whitcher [aut, cre],
           Volker Schmid [aut],
           Andrew Thornton [ctb]
Maintainer Brandon Whitcher <bwhitcher@gmail.com>
Repository CRAN
Date/Publication 2015-04-29 08:13:22

R topics documented:

  dcemriS4-package .................................................. 2
  ADC.fast ............................................................ 3
  aif-models .......................................................... 5
  aifParameters ...................................................... 7
  buckley .............................................................. 8
dcemriS4-package

dcemriS4-package  dcemri: A Package for Medical Image Analysis (S4 implementation)

Description

A collection of routines and documentation that allows one to perform a quantitative analysis of
dynamic contrast-enhanced or diffusion-weighted MRI data. Medical imaging data should be organi-
zed using either the Analyze or NIfTI data formats.

Details

Further information is available in the following vignettes:

dcemriS4  dcemriS4(source, pdf)

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>
Volker Schmid <volkerschmid@users.sourceforge.net>
Andrew Thornton <zeripath@users.sourceforge.net>

References

pharmacokinetic models in dynamic contrast-enhanced magnetic resonance imaging, IEEE Trans-
actions on Medical Imaging, 25 (12), 1627-1636.

the quantitative analysis of dynamic contrast-enhanced MR images based on Bayesian P-splines,
Examples

```r
## Not run:
demo(avg152T1)
demo(avg152T1LR)
demo(avg152T1RL)
demo(buckley)
demo(filtered_func_data)
demo(zstat1)

## End(Not run)
```

---

**Description**

Estimation of apparent diffusion coefficient (ADC) values, using a single exponential function, is achieved through nonlinear optimization.

**Usage**

```r
ADC.fast(dwi, ...)
```

## S4 method for signature 'array'

```r
ADC.fast(dwi, bvalues, dwi.mask, 
control = minpack.lm::nls.lm.control(maxiter = 150), multicore = FALSE, 
verbose = FALSE)
```

```r
adc.lm(signal, b, guess, control = minpack.lm::nls.lm.control())
```

**Arguments**

- `dwi`: Multidimensional array of diffusion-weighted images.
- `...`: Additional variables defined by the method.
- `dwi.mask`: Logical array that defines the voxels to be analyzed.
- `control`: An optional list of control settings for `nls.lm`. See `nls.lm.control` for the names of the settable control values and their effect.
- `multicore`: is a logical variable (default = FALSE) that allows parallel processing via `parallel`.
- `verbose`: Additional information will be printed when `verbose=TRUE`.
- `signal`: Signal intensity vector as a function of b-values.
- `b,bvalues`: Diffusion weightings (b-values).
- `guess`: Initial values of $S_0$ and $D$. 
Details

The `adc.lm` function estimates parameters for a vector of observed MR signal intensities using the following relationship

\[ S(b) = S_0 \exp(-bD) , \]

where \( S_0 \) is the baseline signal intensity and \( D \) is the apparent diffusion coefficient (ADC). It requires the routine `nls.lm` that applies the Levenberg-Marquardt algorithm. Note, low b-values (\(< 50\) or \(< 100\) depending on who you read) should be avoided in the parameter estimation because they do not represent information about the diffusion of water in tissue.

The `ADC.fast` function rearranges the assumed multidimensional (2D or 3D) structure of the DWI data into a single matrix to take advantage of internal R functions instead of loops, and called `adc.lm`.

Value

A list structure is produced with estimates of \( S_0, D \) and information about the convergence of the nonlinear optimization routine.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

References


See Also

`nls.lm`

Examples

```r
S0 <- 10
b <- c(0, 50, 400, 800) # units?
D <- 0.7e-3 # mm^2 / sec (normal white matter)

## Signal intensities based on the (simplified) Bloch-Torry equation
dwi <- function(S0, b, D) {
  S0 * exp(-b*D)
}

set.seed(1234)
signal <- array(dwi(S0, b, D) + rnorm(length(b), sd=0.15),
  c(rep(1,3), length(b)))
ADC <- ADC.fast(signal, b, array(TRUE, rep(1,3)))
```
unlist(ADC) # text output

par(mfrow=c(1,1)) # graphical output
plot(b, signal, xlab="b-value", ylab="Signal intensity")
lines(seq(0,800,10), dwi(seq(0,800,10), D), lwd=2, col=1)
lines(seq(0,800,10), dwi(c(ADC$S0), seq(0,800,10), c(ADC$D)), lwd=2, col=2)
legend("topright", c("True","Estimated"), lwd=2, col=1:2)

---

### aif-models

**Arterial Input Functions**

#### Description

Parametric models for arterial input functions (AIFs) that are compatible with single compartment models for dynamic contrast-enhanced MRI (DCE-MRI).

#### Usage

```r
aif.orton.exp(tt, AB, muB, AG, muG)
orton.exp.lm(tt, aif, guess = c(log(100), log(10), log(1), log(0.1)), nprint=0)
model.orton.exp(tt, aparams, kparams)
```

#### Arguments

- **tt**: is a vector of acquisition times (in minutes) relative to injection of the contrast agent. Negative values should be used prior to the injection.
- **AB, muB, AG, muG**: are parameters of the double exponential function that describe the AIF.
- **aif**: is the vector of observed contrast agent concentrations (data) used to estimate the parametric model.
- **guess**: Initial integer parameter values for the nonlinear optimization.
- **nprint**: is an integer, that enables controlled printing of iterates if it is positive. In this case, estimates of par are printed at the beginning of the first iteration and every `nprint` iterations thereafter and immediately prior to return. If `nprint` is not positive, no tracing information on the progress of the optimization is produced.
- **aparams**: is the vector of parameters \((A_B, \mu_B, A_G, \mu_G)\) associated with the AIF.
- **kparams**: is the vector of parameters \((v_p, K^{trans}, k_{ep})\) associated with the “extended Kety model” for contrast agent concentration.

#### Details

- `aif.orton.exp` displays the exponential AIF from Orton *et al.* (2008) for a known set of AIF parameter values. `model.orton.exp` displays the exponential AIF from Orton *et al.* (2008) for a known set of AIF and compartmental model parameter values. `orton.exp.lm` estimates the AIF parameters, using nonlinear optimization, using a vector of observed contrast agent concentrations.
Value

aif.orton.exp and model.orton.exp return the AIF associated with the pre-specified parameter values.
orton.exp.lm returns a list structure with

- **AB**: The amplitude of the first exponential function.
- **muB**: The decay rate of the first exponential function.
- **AG**: The amplitude of the second exponential function.
- **muG**: The decay rate of the second exponential function.
- **info**: The success (or failure) code from the Levenburg-Marquardt algorithm nls.lm.
- **message**: The text message associated with the info parameters.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

References


See Also

dcemri.lm, extractAIF, nls.lm

Examples

data("buckley")
## Generate AIF params using the orton.exp function from Buckley's AIF
xi <- seq(5, 300, by=5)
time <- buckley$time.min[xi]
aif <- buckley$input[xi]
aifparams <- orton.exp.lm(time, aif)
aifparams$D <- 1
unlist(aifparams[1:4])

aoe <- aif.orton.exp(time, aifparams$AB, aifparams$muB, aifparams$AG, aifparams$muG)
with(buckley, plot(time.min, input, type="l", lwd=2))
lines(time, aoe, lwd=2, col=2)
legend("right", c("Buckley's AIF", "Our approximation"), lty=1,
     lwd=2, col=1:2)
cbind(time, aif, aoe)[1:10,]
Description

Specification of parameters for arterial input functions (AIFs)

Usage

aifParameters(type, user = NULL)

Arguments

type is one of the following character strings associated with an AIF:
  • tofts.kermode
  • fritz.hansen
  • orton.exp
  • orton.cos
  • user
  • empirical

user is a vector of estimated AIF parameters or the empirical AIF values.

Details

See kineticModel for more information.

Value

A vector of parameter values that are appropriate for the model selected.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

See Also

compartmentalModel, dcemri.lm
Simulated Data from Buckley (2002)

Description

In Buckley (2002) tissue residue curves for a Meningioma and a Breast Cancer were simulated using the MMID4 model. Note, the model is described in detail by Bassingthwaighte, J.B. et al. (1984) and Kroll, K et al. (1996). This model accounts for flow dispersion and heterogeneity, and includes capillaries modeled as axially distributed blood-tissue exchange units. A plasma concentration-time curve, AKA arterial input function, was simulated as an input to the model using measurements made by Fritz-Hansen et al. (1996).

Usage

data("buckley")

Format

Two lists are created (breast and meningioma) that contain the simulated time curves and all associated kinetic parameter values.

Source

See below.

References


compartmentalModel

Compartmental Models for Kinetic Parameter Estimation

Description

A selection of parametric models are provided that combine a compartmental model for tissue and a functional form of the arterial input function.

Usage

compartmentalModel(type)

Arguments

type is a character string that identifies the type of compartmental model to be used. Acceptable models include:

"weinmann"  Weinmann AIF convolved with a single compartment (Kety) model
"extended"  Kety model extended with additional vascular compartment (default)
"orton.exp"  Extended model using Orton’s exponential arterial input function
"orton.cos"  Extended model using Orton’s raised cosine arterial input function
"kety.orton.exp"  Kety model using Orton’s exponential arterial input function
"kety.orton.cos"  Kety model using Orton’s raised cosine arterial input function
"weinmann.empirical"  User-specified empirical AIF convolved with a single compartment model
"extended.empirical"  Extended model with user-specified empirical arterial input function

Details

Parametric models from the DCE-MRI literature are provided to the user for kinetic parameter estimation. All models, with the exception of those marked ‘empirical’ incorporate a parametric model for the arterial input function (AIF).

Value

A function.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

See Also

aifParameters, dcemri.bayes, dcemri.lm, dcemri.map
convFFT

Convolution of 3D Arrays using the Fourier Transform

Description

Convolve a three-dimensional array with another three-dimensional array using the Fast Fourier Transform (FFT).

Usage

convFFT(A, B, C, FFTA = NULL)

Arguments

A is a three-dimensional array (“the template”).
B is a three-dimensional array (“the target”).
C is a vector of length three (the center of “the template”).
FFTA is the three-dimensional Fourier transform of A, this may save time when looping over multiple “targets”.

Details

The arrays A and B are transformed into the Fourier domain and multiplied together (equivalent to a convolution in the image domain across all spatial locations simultaneously).

Value

A three-dimensional array, the same dimension as the input arrays, that is the convolution of the “target” to the “template” at all spatial locations.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

References


See Also

fft, fastTemplateMatching, shift3D
Examples

cube <- array(0, c(20, 20, 1))
cube[9:12, 9:12, 1] <- 1
tkernel <- array(0, c(20, 20, 1))
tkernel[, , 1] <- c(.5, 1, .5, rep(0, 20-3))
tcenter <- findCenter(ifelse(tkernel > 0, TRUE, FALSE))
out <- convFFT(tkernel, cube, tcenter)
out[8:13, 8:13, 1] # text output

par(mfrow=c(2,2)) # graphic output
image(drop(tkernel), col=oro.nifti::tim.colors(), main="Template")
image(drop(cube), col=oro.nifti::tim.colors(), main="Target")
image(drop(out), col=oro.nifti::tim.colors(), main="Output")

dcemri.bayes
Bayesian Methods for Pharmacokinetic Modeling of Dynamic Contrast-Enhanced MRI Data

Description

Bayesian analysis of contrast agent concentration time curves from DCE-MRI.

Usage

dcemri.bayes(conc, ...)  
## S4 method for signature 'array'

dcemri.bayes(conc, time, img.mask, model = "extended",  
aif = NULL, user = NULL, nriter = 3000, thin = 3, burnin = 1000,  
tune = 267, ab.ktrans = c(0, 1), ab.kep = ab.ktrans, ab.vp = c(1, 19),  
ab.tauepsilon = c(1, 1/1000), samples = FALSE, multicore = FALSE,  
verbose = FALSE, dic = FALSE, ...)

Arguments

conc    Matrix or array of concentration time series (last dimension must be time).
...     Additional parameters to the function.
time    Time in minutes.
img.mask Mask matrix or array. Voxels with mask=0 will be excluded.
model   is a character string that identifies the type of compartmental model to be used.  
Acceptable models include:
"weinmann" Tofts & Kermode AIF convolved with single compartment model  
"extended" Weinmann model extended with additional vascular compartment  
(default)  
"orton.exp" Extended model using Orton’s exponential AIF  
"kety.orton.exp" Kety model using Orton’s exponential AIF
aif is a character string that identifies the parameters of the type of arterial input function (AIF) used with the above model. Acceptable values are: tofts.kermode (default) or fritz.hansen for the weinmann and extended models; orton.exp (default) or user for the orton.exp and kety.orton.exp model.

user Vector of AIF parameters. For Tofts and Kermode: $a_1, m_1, a_2, m_2$; for Orton et al.: $A_b, \mu_b, A_g, \mu_g$.

nriters Total number of iterations.

thin Thining factor.

burnin Number of iterations for burn-in.

tune Number for iterations for tuning. The algorithm will be tuned to an acceptance rate between 0.3 and 0.6.

ab.ktrans Mean and variance parameter for Gaussian prior on $\log(K^{\text{trans}})$.

ab.kep Mean and variance parameter for Gaussian prior on $\log(k_{ep})$.

ab.vp Hyper-prior parameters for the Beta prior on $v_p$.

ab.tauepsilon Hyper-prior parameters for observation error Gamma prior.

samples If TRUE output includes samples drawn from the posterior distribution for all parameters.

multicore If TRUE algorithm is parallelized using multicore.

verbose Logical variable (default = FALSE) that allows text-based feedback during execution of the function.

dic If TRUE, the deviance information criterion (DIC) and effective number of parameters (pD) will be computed. If "samples = TRUE", then samples of the DIC and pD will be given.

vp Fractional occupancy in the plasma space.

Details
See Schmid et al. (2006) for details.

Value
Parameter estimates and their standard errors are provided for the masked region of the multidimensional array. All multi-dimensional arrays are output in nifti format. They include:

ktrans Transfer rate from plasma to the extracellular, extravascular space (EES).

ktranserror Error on $K^{\text{trans}}$.

kep Rate parameter for transport from the EES to plasma.

keperror Error on $k_{ep}$.

ve Fractional occupancy by EES (the ratio between ktrans and kep).

verror Error on $v_e$.

vp Fractional occupancy by plasma.

sigma2 The residual sum-of-squares from the model fit.
time  Acquisition times (for plotting purposes).
DIC   Deviance information criterion.
DIC.map Contribution to DIC per voxel.
pD    Effective number of parameters.
pD.map Contribution to pD per voxel.

Note, not all parameters are available under all models choices.

Author(s)
Volker Schmid <volkerschmid@users.sourceforge.net>

References

See Also
dcemri.lm, dcemri.map, dcemri.spline

Examples

```r
data("buckley")
xi <- seq(5, 300, by=5)
img <- array(t(breast$data)[,xi], c(13,1,1,60))
mask <- array(TRUE, dim(img)[1:3])
time <- buckley$time.min[xi]

## Bayesian estimation with Fritz-Hansen default AIF
fit.bayes <- dcemri.bayes(img, time, mask, aif="fritz.hansen",
nriters=1000, thin=2, burnin=200)

## Bayesian estimation with "orton.exp" function fit to Buckley's AIF
aif <- buckley$input[xi]
aifparams <- orton.exp.lm(time, aif)
aifparams$D <- 1
fit.bayes.aif <- dcemri.bayes(img, time, mask, model="orton.exp",
aif="user", user=aifparams,
nriters=1000, thin=2, burnin=200)

plot(breast$ktrans, fit.bayes$ktrans, xlim=c(0,1), ylim=c(0,1),
     xlab=expression("True \; K^{\{trans\}}"),
     ylab=expression("Estimated \; K^{\{trans\}} \; (Bayesian)"))
points(breast$ktrans, fit.bayes.aif$ktrans, pch=2)
abline(0, 1, lwd=2, col=2)
legend("right", c("extended/fritz.hansen","orton.exp/user"), pch=1:2)

fit.lm <- dcemri.lm(img, time, mask, aif="fritz.hansen"
fit.lm.aif <- dcemri.lm(img, time, mask, model="orton.exp", aif="user",
```

Parameter estimation for single compartment models is performed using literature-based or user-specified arterial input functions. The Levenburg-Marquardt algorithm does the heavy lifting.

### Usage

dcemri.lm(conc, ...)  

```
# S4 method for signature 'array'
dcemri.lm(conc, time, img.mask, model = "extended",  
aif = NULL, control = minpack.lm::nls.lm.control(), user = NULL,  
guess = NULL, multicore = FALSE, verbose = FALSE, ...)```

### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conc</td>
<td>is a multidimensional (1D-4D) array of contrast agent concentrations. The last dimension is assumed to be temporal, while the previous dimensions are assumed to be spatial.</td>
</tr>
<tr>
<td>time</td>
<td>is a vector of acquisition times (in minutes) relative to injection of the contrast agent. Negative values should be used prior to the injection.</td>
</tr>
<tr>
<td>img.mask</td>
<td>is a (logical) multidimensional array that identifies the voxels to be analyzed. Has to have same dimension as conc minus temporal dimension.</td>
</tr>
<tr>
<td>mode</td>
<td>is a character string that identifies the type of compartmental model to be used. Acceptable models include: &quot;weinmann&quot; Tofts &amp; Kermode AIF convolved with single compartment model &quot;extended&quot; Weimann model extended with additional vascular compartment (default)</td>
</tr>
</tbody>
</table>

---

dcemri.lm

*Pharmacokinetic Models for Dynamic Contrast-Enhanced MRI Data*
"orton.exp" Extended model using Orton’s exponential AIF
"orton.cos" Extended model using Orton’s raised cosine AIF
"kety.orton.exp" Kety model using Orton’s exponential AIF
"kety.orton.cos" Kety model using Orton’s raised cosine AIF

aif is a character string that identifies the parameters of the type of arterial input function (AIF) used with the above model. Acceptable values are:

- tofts.kermode (default) for the weinmann and extended models
- fritz.hansen for the weinmann and extended models
- “empirical” for the weinmann and extended models
- orton.exp (default) for the orton.exp and kety.orton.exp model
- orton.cos (default) for the orton.cos and kety.orton.cos model.
- user for the orton.exp and orton.cos model.

All AIF models set the parametric form and parameter values – except user, where a set of user-defined parameter values are allowed, and empirical, where a vector of values that fully characterize the empirical AIF.

control is a list of parameters used by nls.lm.control that are set by default, but may be customized by the user.

user is a list with the following parameters required: D, AB, muB, AG, muG.

guess is a vector of starting values for kinetic parameter estimation. The vector must have length = 3 (with names th0, th1 and th3) when the extended Kety model is used with the vascular parameter and length = 2 (with names th1 and th3) otherwise.

multicore is a logical variable (default = FALSE) that allows parallel processing via parallel.

verbose is a logical variable (default = FALSE) that allows text-based feedback during execution of the function.

Details

Compartmental models are the solution to the modified general rate equation (Kety 1951). The specific parametric models considered here include the basic Kety model

\[ C_i(t) = K^{\text{trans}} \left[ C_p(t) \otimes \exp(-k_{ep}t) \right], \]

where \( \otimes \) is the convolution operator, and the so-called extended Kety model

\[ C_i(t) = u_p C_p(t) + K^{\text{trans}} \left[ C_p(t) \otimes \exp(-k_{ep}t) \right]. \]

The arterial input function must be either literature-based (with fixed parameters) or the exponential AIF from Orton et al. (2008) with user-defined parameters.

Value

Parameter estimates and their standard errors are provided for the masked region of the multidimensional array. All multi-dimensional arrays are provided in nifti format. They include:

ktrans Transfer rate from plasma to the extracellular, extravascular space (EES).
kep  Rate parameter for transport from the EES to plasma.
ve  Fractional occupancy by EES (the ratio between $K^{\text{trans}}$ and $k_{ep}$).
vp  Fractional occupancy in the plasma space.
ktranserror  Standard error for $K^{\text{trans}}$.
keperror  Standard error for $k_{ep}$.
v perror  Standard error for $v_p$.

The residual sum-of-squares is also provided, along with the original acquisition times (for plotting purposes).

Note
WARNING: when using the empirical AIF, a linear interpolation is used to upsample the AIF to a one-second sampling rate. This allows one to utilize a computationally efficient numeric method to perform the convolution. If the empirical AIF is sampled faster than one Hertz, then the upsampling operation will become a downsampling operation. This should not have any serious effect on the parameter estimates, but caution should be exercised if very fast sampling rates are used to obtain an empirical AIF.

Author(s)
Brandon Whitcher <bwhitcher@gmail.com>,
Volker Schmid <volkerschmid@users.sourceforge.net>

References

See Also
dcemri.bayes, dcemri.map, dcemri.spline, nls.lm

Examples

```r
data("buckley")

## Empirical arterial input function
img <- array(t(breast$data), c(13,1,301))
time <- buckley$time.min
mask <- array(TRUE, dim(img)[1:3])

## Estimate kinetic parameters directly from Buckley's empirical AIF
fit1 <- dcemri.lm(img, time, mask, model="weinmann", aif="empirical",
                  user=buckley$input)
fit2 <- dcemri.lm(img, time, mask, model="extended", aif="empirical",
                  user=buckley$input)

## Set up breast data for dcemri
xi <- seq(5, 300, by=3)
img <- array(t(breast$data)[,xi], c(13,1,length(xi))}
time <- buckley$time.min[1]
input <- buckley$input[1]

## Generate AIF params using the orton.exp function from Buckley's AIF
(aifparams <- orton.exp.lm(time, input))
fit3 <- dcemri.lm(img, time, mask, model="orton.exp", aif="user",
                  user=aifparams)

## Scatterplot comparing true and estimated \( k_{\text{trans}} \) values
plot(breast$ktrans, fit1$ktrans, xlab=paste("True \( k_{\text{trans}} \)"),
     ylab=paste("Estimated \( k_{\text{trans}} \)"),
     xlim=c(0,0.75), ylim=c(0,0.75),
     pch=2)
points(breast$ktrans, fit2$ktrans, pch=3)
abline(0, 1, lwd=1.5, col=2)
legend("bottomright", c("weinmann/empirical", "extended/empirical",
                         "orton.exp/user"), pch=1:3)
cbind(breast$ktrans, fit1$ktrans[,1], fit2$ktrans[,1], fit3$ktrans[,1])

## Scatterplot comparing true and estimated \( v_p \) values
plot(breast$v$p, fit1$v$p, type="n", xlab=paste("True \( v_p \)"),
     ylab=paste("Estimated \( v_p \)"),
     xlim=c(0,0.15), ylim=c(0,0.15),
     pch=2)
points(breast$v$p, fit2$v$p, pch=3)
abline(0, 1, lwd=1.5, col=2)
```

Description

Maximum-a-posteriori (MAP) estimation for single compartment models is performed using literature-based or user-specified arterial input functions.

Usage

dcemri.map(conc, ...)  

## S4 method for signature 'array'
dcemri.map(conc, time, img.mask, model = "extended",  
aif = NULL, user = NULL, ab.ktrans = c(0, 1), ab.kep = ab.ktrans,  
ab.vp = c(1, 19), ab.tauepsilon = c(1, 1/1000), maxit = 5000,  
samples = FALSE, multicores = FALSE, verbose = FALSE, ...)  

Arguments

conc Matrix or array of concentration time series (last dimension must be time).
... Additional parameters to the function.
time Time in minutes.
img.mask Mask matrix or array. Voxels with mask=0 will be excluded.
model is a character string that identifies the type of compartmental model to be used. Acceptable models include:
   "weinmann" Tofts & Kermode AIF convolved with single compartment model
   "extended" Weinmann model extended with additional vascular compartment
                   (default)
   "orton.exp" Extended model using Orton’s exponential AIF
   "kety.orton.exp" Kety model using Orton’s exponential AIF
   "orton.cos" Extended model using Orton’s raised cosine AIF
   "kety.orton.cos" Kety model using Orton’s raised cosine AIF
aif is a character string that identifies the parameters of the type of arterial input  
function (AIF) used with the above model. Acceptable values are: tofts.kermode  
(default) or fritz.hansen for the weinmann and extended models; orton.exp  
(default) or user for the orton.exp model and orton.exp model; user for the  
orton.cos model and orton.cos model.
user Vector of AIF parameters. For Tofts and Kermode: a1, m1, a2, m2; for Orton et  
el.: A_b, mu_b, A_g, mu_g.
ab.ktrans Mean and variance parameter for Gaussian prior on log(K^trans).
ab kep  Mean and variance parameter for Gaussian prior on log(kep).
ab.vp Hyper-prior parameters for the Beta prior on vp.
ab.tauepsilon Hyper-prior parameters for observation error Gamma prior.
maxit The maximum number of iterations for the optimization procedure.
samples If TRUE output includes samples drawn from the posterior distribution for all parameters.
multicore If TRUE algorithm is parallelized using multicore.
verbose Logical variable (default = FALSE) that allows text-based feedback during execution of the function.

Details

Implements maximum a posteriori (MAP) estimation for the Bayesian model in Schmid et al. (2006).

Value

Parameter estimates and their standard errors are provided for the masked region of the multidimensional array. The multi-dimensional arrays are provided in nifti format. They include:

ktrans Transfer rate from plasma to the extracellular, extravascular space (EES).
kep Rate parameter for transport from the EES to plasma.
ve Fractional occupancy by EES (the ratio between ktrans and kep).
vp Fractional occupancy by plasma.
sigma2 The residual sum-of-squares from the model fit.
time Acquisition times (for plotting purposes).

Note, not all parameters are available under all models choices.

Author(s)

Volker Schmid <volkerschmid@users.sourceforge.net>

References


See Also
dcemri.lm, dcemri.bayes
**Examples**

```r
data("buckley")
xi <- seq(5, 300, by=5)
img <- array(t(breast$data)[,xi], c(13,1,60))
mask <- array(TRUE, dim(img)[1:3])
time <- Buckley$time.min[xi]

## MAP estimation with extended Kety model and Fritz-Hansen default AIF
fit.map.vp <- dcemri.map(img, time, mask, aif="fritz.hansen")

## Nonlinear regression with extended Kety model and Fritz-Hansen default AIF
fit.lm.vp <- dcemri.lm(img, time, mask, aif="fritz.hansen")

plot(breast$ktrans, fit.map.vp$ktrans, xlim=c(0,1), ylim=c(0,1),
     xlab=expression(paste("True",",", K[\{trans\}]),
     ylab=expression(paste("Estimated",",", K[\{trans\}]))
points(breast$ktrans, fit.lm.vp$ktrans, pch=3)
abline(0,1, lwd=2, col=2)
legend("bottomright", c("MAP Estimation (fritz.hansen)",
                         "Levenburg-Marquardt (fritz.hansen)",
                         pch=c(1,3))

## MAP estimation with Kety model and Fritz-Hansen default AIF
fit.map <- dcemri.map(img, time, mask, model="weinmann", aif="fritz.hansen")

## Nonlinear regression with Kety model and Fritz-Hansen default AIF
fit.lm <- dcemri.lm(img, time, mask, model="weinmann", aif="fritz.hansen")

cbind(breast$kep, fit.lm$kep[,1], fit.lm.vp$kep[,1], fit.map$kep[,1],
       fit.map.vp$kep[,1])

summary(breast$ktrans, fit.lm.ktrans[,1], fit.lm.vp$ktrans[,1],
        fit.map.ktrans[,1], fit.map.vp$ktrans[,1])

```

---

**dcemri.spline**  
*Bayesian P-Splines for Dynamic Contrast-Enhanced MRI Data*

**Description**

Quantitative analysis of DCE-MRI typically involves the convolution of an arterial input function (AIF) with a nonlinear pharmacokinetic model of the contrast agent concentration. This function takes a semi-parametric penalized spline smoothing approach, with which the AIF is convolved with a set of B-splines to produce a design matrix using locally adaptive smoothing parameters based on Bayesian penalized spline models (P-splines).

**Usage**

```r
dcemri.spline(conc, ...)

## S4 method for signature 'array'
dcemri.spline(conc, time, img.mask, time.input = time,
              model = "weinmann", aif = "tofts.kermode", user = NULL,
              aif.observed = NULL, nriters = 500, thin = 5, burnin = 100,
```
Arguments

- `conc`: Matrix or array of concentration time series (last dimension must be time).
- `time`: Time in minutes.
- `img.mask`: Mask matrix or array. Voxels with mask = 0 will be excluded.
- `time.input`: Time in minutes for observed arterial input function (default = 'time').
- `model`: Only if `nlr = TRUE` Response model fitted to the estimated response function. Acceptable values include: "AATH" or "weinmann" (default).
- `aif`: is a character string that identifies the parameters of the arterial input function. Acceptable values are: tofts.kermode, fritz.hansen or observed. If observed you must provide the observed concentrations in `aif.observed`.
- `ab.hyper`: Hyper priors for adaptive smoothness parameter 
- `ab.tau.epsilon`: Hyper-prior parameters for observation error Gamma prior.
- `k`: Order of B-Splines.
- `p`: Number of knots of B-Spline basis.
- `rw`: Order of random walk prior. Acceptable values are 1 and 2.
- `knots`: Vector of knots. Use this if you need unequally spaced knots.
- `nlr`: If TRUE, a response model is fitted to the estimated response function.
- `t0.compute`: If TRUE, the onset time will be estimated from response function.
- `samples`: If TRUE output includes samples drawn from the posterior distribution for all parameters.
- `multicore`: (logical) use the parallel package.
- `verbose`: (logical) allows text-based feedback during execution of the function (default = FALSE).
- `response`: If TRUE, the response functions per voxel are returned.
- `fitted`: If TRUE, then fitted time curved per voxel are returned.

Details

See Schmid et al. (2009) for more details.
Value

The maximum of the response function $f_p$ for the masked region is provided by default. Where appropriate, response functions, fitted functions, and parameter estimates (along with their standard errors) are provided. All multi-dimensional arrays are provided in nifti format.

Author(s)

Volker Schmid <volkerschmid@users.sourceforge.net>

References


See Also
dcemri.bayes, dcemri.lm, dcemri.map

Examples

data("buckley")
xi <- seq(5, 300, by=5)
img <- array(t(breast$data)[,xi], c(13,1,1,60))
mask <- array(TRUE, dim(img)[1:3])
time <- buckley$time.min[xi]

## Generate AIF params using the orton.exp function from Buckley's AIF
aif <- buckley$input[xi]

fit.spline <- dcemri.spline(img, time, mask, aif="fritz.hansen",
                           nriter=125, thin=3, burnin=25, nlr=TRUE)
fit.spline.aif <- dcemri.spline(img, time, mask, aif="observed",
                               aif.observed=aif, nriter=125, thin=3,
                               burnin=25, nlr=TRUE)

plot(breast$ktrans, fit.spline$ktrans, xlab=expression(paste("Estimated ", K^{trans})),
     ylab=expression(paste("Estimated ", K^{trans})))
points(breast$ktrans, fit.spline.aif$ktrans, pch=2)
abline(0, 1, lwd=1.5, col="red")
legend("right", c("fritz.hansen", "observed"), pch=1:2)

**doubleAngleMethod**

*Double-Angle Method for B1+ Mapping*
**doubleAngleMethod**

**Description**
For in vivo MRI at high field ($\geq 3$ T) it is essential to consider the homogeneity of the active B1 field (B1+). The B1+ field is the transverse, circularly polarized component of B1 that is rotating in the same sense as the magnetization. When exciting or manipulating large collections of spins, nonuniformity in B1+ results in nonuniform treatment of spins. This leads to spatially varying image signal and image contrast and to difficulty in image interpretation and image-based quantification.

**Usage**
```python
doubleAngleMethod(low, high, low_deg)
```

**Arguments**
- `low` is the (3D) array of signal intensities at the low flip angle.
- `high` is the (3D) array of signal intensities at the high flip angle (note, $2 \times$ low = high).
- `low_deg` is the low flip angle (in degrees).

**Details**
The proposed method uses an adaptation of the double angle method (DAM). Such methods allow calculation of a flip-angle map, which is an indirect measure of the B1+ field. Two images are acquired: $I_1$ with prescribed tip $\alpha_1$ and $I_2$ with prescribed tip $\alpha_2 = 2\alpha_1$. All other signal-affecting sequence parameters are kept constant. For each voxel, the ratio of magnitude images satisfies

\[
\frac{I_2(r)}{I_1(r)} = \frac{\sin \alpha_2(r) f_2(T_1, TR)}{\sin \alpha_1(r) f_1(T_1, TR)}
\]

where $r$ represents spatial position and $\alpha_1(r)$ and $\alpha_2(r)$ are tip angles that vary with the spatially varying B1+ field. If the effects of $T_1$ and $T_2$ relaxation can be neglected, then the actual tip angles as a function of spatial position satisfy

\[
\alpha(r) = \arccos \left( \frac{1}{2} \sqrt{\frac{I_2(r)}{2I_1(r)}} \right)
\]

A long repetition time ($TR \leq 5T_1$) is typically used with the double-angle methods so that there is no $T_1$ dependence in either $I_1$ or $I_2$ (i.e., $f_1(T_1, TR) = f_2(T_1, TR) = 1.0$). Instead, the proposed method includes a magnetization-reset sequence after each data acquisition with the goal of putting the spin population in the same state regardless of whether the or $\alpha_2$ excitation was used for the preceding acquisition (i.e., $f_1(T_1, TR) = f_2(T_1, TR) \neq 1.0$).

**Value**
An array, the same dimension as the acquired signal intensities, is returned containing the multiplicative factor associated with the low flip angle acquisition. That is, if no B1+ inhomogeneity was present then the array would only contain ones. Numbers other than one indicate the extent of the inhomogeneity as a function of spatial location.

**Author(s)**
Brandon Whitcher <bwhitcher@gmail.com>
References


---

**expConv**  
Convolution of Exponential Functions

**Description**

...

**Usage**

expConv(input, k1, k2)

**Arguments**

- `input`...
- `k1`...
- `k2`...

**Details**

...

**Value**

The convolved time series.

**Author(s)**

Brandon Whitcher <bwhitcher@gmail.com>

---

**extractAIF**  
Seed Growing for a 4D Array

**Description**

Seed growing algorithm to find voxels in a three-dimensional array according to their correlation to a seed voxel. The correlation is measured according to the fourth dimension of the array.

**Usage**

extractAIF(img, x, y, z, thresh = 0.9)
**fastTemplateMatching**

**Arguments**

- `img` is the four-dimensional array of medical imaging data.
- `x, y, z` are the coordinates of the seed voxel.
- `thresh` is the minimum correlation for inclusion in the region.

**Details**

Correlation coefficients are computed for every voxel in the input array. A recursive algorithm is then used to grow the region of interest (ROI) from the seed voxel in three dimensions. All adjacent voxels, where the correlation exceeds the threshold, are included.

**Value**

- `coord` is a matrix of the three-dimensional coordinates \((x, y, z)\) for all voxels found by the algorithm.
- `conc` is a matrix whose rows correspond to the voxels found by the algorithm and whose columns are the fourth dimension from the input array (e.g., contrast agent concentration time curve).
- `mask` is an array of boolean values, where only voxels included by the algorithm are given a value greater than zero.
- `cor` is an array that mimics the mask, but contains the estimated correlation coefficients for all voxels included by the algorithm.

**Author(s)**

Volker Schmid <volker.schmid@users.sourceforge.net>

---

**Description**

Motion correction and/or co-registration of three-dimensional arrays (medical imaging data) are performed by applying a user-defined mask of voxels. Normalized cross-correlations (in 3D) are computed using the FFT.

**Usage**

```r
fastTemplateMatching(input, ...)  
# S4 method for signature 'array'
fastTemplateMatching(input, ...)
```
Arguments

input is a four-dimensional array of signal intensities.
... Additional variables passed to the plot function.

Details

An extremely basic method of motion correction/co-registration is implemented by estimating “local” cross-correlations based on a binary mask that is a subset of the original three-dimensional volume. All convolutions are preformed via the FFT (fft) and repetitive calculations are minimized where possible.

Only whole-voxel translations are considered. This does not begin to capture the true effects of motion in soft tissue, but we assume that the object of interest (e.g., tumor) is a fairly rigid structure. Potential extensions include rigid-body, affine and nonlinear registration techniques along with interpolation schemes in order to capture intra-voxel manipulations of the data.

Value

A list of objects are returned:

out Motion-corrected version of the four-dimensional array.
offset Translations (in 3D) for each volume in the 4D array.
t.center Estimated center of the binary mask.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

References

www.idiom.com/~zilla/

See Also

convFFT, findCenter, shift3D
Arguments

\( M \) is a binary mask (multidimensional array of logical values).

Details

This method most likely only works with convex three-dimensional shapes (e.g., a hyper-rectangle). Further testing is required to know the limits of the current implementation.

Value

A vector of values the same length as the input array.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

See Also

fastTemplateMatching

Examples

\[
\begin{align*}
M & \leftarrow \text{array} (\text{FALSE, rep(10, 3)}) \\
M[6:10, 6:10, 6:10] & \leftarrow \text{TRUE} \\
M_c & \leftarrow \text{findCenter} (M) \\
\text{print} (M_c)
\end{align*}
\]

kineticModel  Pharmacokinetic Models

Description

Kinetic curves from single compartment models are computed from kinetic parameters.

Usage

kineticModel(time, par, model = "extended", aif = "fritz.hansen")

Arguments

time is a vector of acquisition times (in minutes).
par is a list of kinetic parameters; e.g., \texttt{list("ktrans"=0.5,"kep"=1)}.
model is a character string that identifies the type of compartmental model to be used. Acceptable models include: “weinmann” Tofts & Kermode AIF convolved with single compartment model “extended” (default) Weinmann model extended with additional vascular compartment, ...
aif is a character string that identifies the type of arterial input function (AIF) to be used. Acceptable AIF models include: tofts.kermod, fritz.hansen
Details

Compartmental models are the solution to the modified general rate equation (Kety 1951). The specific parametric models considered here include the basic Kety model

$$C_t(t) = K^{trans} [C_p(t) \odot \exp(-k_{ep} t)],$$

where $\odot$ is the convolution operator, and the so-called extended Kety model

$$C_t(t) = v_p C_p(t) + K^{trans} [C_p(t) \odot \exp(-k_{ep} t)].$$

The arterial input function must be literature-based (with fixed parameters).

Value

Computed pharmacokinetic curve.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com> and Volker Schmid <volkerschmid@users.sourceforge.net>

References


See Also

dcemri.lm, dcemri.bayes, dcemri.spline

Examples

data("buckley")
x1 <- seq(5, 300, by=5)
img <- array(t(breast$data[,x1]), c(13,1,60)) mask <- array(TRUE, dim(img)[1:3]) time <- buckley$time.min[x1]

fit.lm <- dcemri.lm(img, time, mask, aif="fritz.hansen")
par.lm <- c("vp"=fit.lm$vp[3], "ktrans"=fit.lm$ktrans[3], "kep"=fit.lm$kep[3])
curve.lm <- kineticModel(time, par.lm)
plot(time, img[3,1,1], xlab="time", ylab="contrast agent concentration")
lines(time, curve.lm, lwd=2, col=2)

fit.bayes <- dcemri.bayes(img, time, mask, aif="fritz.hansen")
par.bayes <- c("vp"=fit.bayes$v[3], "ktrans"=fit.bayes$ktrans[3],
        "kep"=fit.bayes$kep[3])
curve.bayes <- kineticModel(time, par.bayes)
lines(time, curve.bayes, lwd=2, col=4)
legend("bottomright", c("Levenburg-Marquardt (extended/fritz.hansen)",
            "Bayesian Estimation (extended/fritz-hansen)"),
            lwd=2, col=c(2,4))
cbind(time, img[3,3,], curve.lm, curve.bayes)[20:30,]

---

R10.lm

Estimate Intrinsic Tissue Relaxivity

Description

Estimation of the intrinsic tissue relaxivity is achieved through nonlinear optimization and the dynamic signal intensities are converted into contrast agent concentration.

Usage

R10.lm(signal, alpha, TR, guess, control = minpack.lm::nls.lm.control())

E10.lm(signal, alpha, guess, control = minpack.lm::nls.lm.control())

R1.fast(flip, ...)

## S4 method for signature 'array'
R1.fast(flip, flip.mask, fangles, TR,
    control = minpack.lm::nls.lm.control(), multicore = FALSE,
    verbose = FALSE)

CA.fast(dynamic, ...)

## S4 method for signature 'array'
CA.fast(dynamic, dyn.mask, dangle, flip, fangles, TR,
    r1 = 4, control = minpack.lm::nls.lm.control(maxiter = 200),
    multicore = FALSE, verbose = FALSE)

CA.fast2(dynamic, ...)

## S4 method for signature 'array'
CA.fast2(dynamic, dyn.mask, dangle, flip, fangles, TR,
    r1 = 4, verbose = FALSE)
Arguments

- **signal**: the vector of signal intensities as a function of flip angles.
- **alpha**: the vector of flip angles (in degrees).
- **TR**: is the relaxation time (in seconds) used in the acquisition of the MRI data.
- **guess**: is the vector of initial values for the parameters of interest: $m_0$ and $R_{10}$.
- **control**: An optional list of control settings for `nls.lm`. See `nls.lm.control` for the names of the settable control values and their effect.
- **flip**: a multidimensional array of contrast agent concentrations. The last dimension is assumed to be a function of the flip angles, while the previous dimensions are assumed to be spatial.
- **flip.mask, dyn.mask**: is a (logical) multidimensional array that identifies the voxels to be analyzed.
- **fangles**: is the vector of flip angles (in degrees).
- **multicore**: is a logical variable (default = FALSE) that allows parallel processing via parallel.
- **verbose**: is a logical variable (default = FALSE) that allows text-based feedback during execution of the function.
- **dynamic**: a multidimensional array of contrast agent concentrations. The last dimension is assumed to be temporal, while the previous dimensions are assumed to be spatial.
- **dangle**: is the flip angle used to acquire the dynamic MRI data.
- **r1**: is the spin-lattice relaxivity constant (default = 4.39 for 1.5T). For 3T data it may be necessary to adjust this value.

Details

The `E10.lm` and `R10.lm` functions estimate parameters for a vector of observed MR signal intensities, as a function of flip angle, using the following relationship

\[
S(\alpha) = m_0 \sin(\alpha) \left( 1 - \exp(-\text{TR}/T_1) \right) \left( 1 - \cos(\alpha) \exp(-\text{TR}/T_1) \right). 
\]

The only difference between the two functions is exactly what is being estimated in the nonlinear least squares formulation. They both require the function `nls.lm` that uses the Levenberg-Marquardt algorithm.

The `CA.fast` function calls on `R1.fast` to rearrange the assumed multidimensional (2D or 3D) structure of the multiple flip-angle data into a single matrix to take advantage of internal R functions instead of loops when calling `E10.lm`. Conversion of the dynamic signal intensities to contrast agent concentration is performed via

\[
[Gd] = \frac{1}{r_1} \left( \frac{1}{T_1} - \frac{1}{T_{10}} \right).
\]

The `CA2.fast` function assumes only two flip angles have been acquired and uses an approximation to the nonlinear relationship between signal intensity and flip angle to enable conversion from signal intensity to contrast agent concentration.
Value

A list structure is produced with (all or some of the) parameter estimates:

- $M_0$: Scaling factor between signal intensity and T1.
- $R_{10}$: Pre-injection tissue relaxation rate (3D array); $R_{10} = 1/T_{10}$.
- $R_{1t}$: Time-varying tissue relaxation rate (4D array); $R_{1}(t) = 1/T_{1}(t)$.
- conc: Contrast agent concentration (4D array).

and information about the convergence of the nonlinear optimization routine.

Note

The longitudinal relaxivity is set, by default, to $r_1 = 4(MM \cdot s)^{-1}$ which is a reasonable value for gadolinium contrast agents at 1.5 Tesla. Double-check the scanning procedure manual to ensure the correct value is used.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

References


See Also

dcemri.lm, nls.lm

Examples

```r
## Parameters for simulated data
S0 <- 100
TR <- 5 / 1000  # seconds
T1 <- 1.5  # seconds
alpha <- seq(2, 24, by=2)  # degrees

## Signal intensities for spoiled gradient echo image
gre <- function(S0, TR, T1, alpha) {
  theta <- alpha * pi/180  # radians
  ...}
```


S0 * (1 - exp(-TR/T1)) * sin(theta) / (1 - cos(theta) * exp(-TR/T1))

```r
set.seed(1234)
signal <- array(gre(S0, TR, T1, alpha) + rnorm(length(alpha), sd=.15),
c(rep(1,3), length(alpha)))
out <- R1.fast(signal, array(TRUE, rep(1,3)), alpha, TR)
unlist(out)
plot(alpha, signal, xlab="Flip angle", ylab="Signal intensity")
lines(alpha, gre(S0, TR, T1, alpha), lwd=2, col=1)
lines(alpha, gre(c(out$M0), TR, 1/c(out$R10), alpha), lwd=2, col=2)
legend("topright", c("True","Estimated"), lwd=2, col=1:2)
```

### Description

Quantification of relative cerebral blood volume (rCBV) using the first pass from a bolus injection of a contrast agent.

### Usage

```r
rcBV.fast(signal, ...)  
## S4 method for signature 'array'
rcBV.fast(signal, mask, aif, time, multicore = FALSE,
          verbose = FALSE)

rcBV(Ct, Ca, time, Hf = 1, rho = 1)
```

### Arguments

- **signal** is a multidimensional array of signal intensities (or concentrations). The last dimension is assumed to be a function of the acquisition times, while the previous dimensions are assumed to be spatial.
- **...** Additional variables defined by the method.
- **mask** is a (logical) multidimensional array that identifies the voxels to be analyzed.
- **aif** Arterial Input Function.
- **time** is the vector of acquisition times associated with the dynamic data.
- **multicore** is a logical variable (default = FALSE) that allows parallel processing via `parallel`.
- **verbose** is a logical variable (default = FALSE) that allows text-based feedback during execution of the function.
- **Ct** is the time series of contrast agent concentration in tissue.
- **Ca** is the time series of contrast agent concentration in the blood.
- **Hf** is the hematocrit factor.
- **rho** is the density of brain tissue.
Value

A nifti object containing the estimates of regional cerebral blood volume (rCBV).

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

shift3D Shift a 3D Array in One Dimension

Description

One axis of the three-dimensional array is translated by an integer amount. This is useful when applying convolution operators in the Fourier domain.

Usage

shift3D(A, s, type, fill = 0)

Arguments

A is a three-dimensional array.

s is the integer number of translation steps.

type is a character string using anatomical coordinates assuming a transverse acquisition scheme ("LR" = left-right = x-axis, "AP" = anterior-posterior = y-axis, "SI" = superior-inferior = z-axis).

fill is the quantity used to fill gaps induced by the translations (circular boundary conditions are NOT used).

Value

A three-dimensional array is returned, the same dimension as the original array, with one dimension translated.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

See Also

cConvFFT
Examples

cube <- array(0, rep(20,3))
cube[9:12,9:12,9:12] <- 1
cube.shift <- shift3D(cube, 5, type="AP")
par(mfrow=c(1,2), mar=rep(0.5,4))
image(cube[,10], xlab="", ylab="", axes=FALSE)
image(cube.shift[,10], xlab="", ylab="", axes=FALSE)

T2.fast  Quantitative T2 Methods

Description

The regional blood volume is found by integrating of the tissue concentration curve and the arterial input function (AIF). In order to avoid reperfusion effects on the rCBV measurements, the tissue and arterial concentration curves must first be reduced to their first-pass versions.

Usage

T2.fast(cpmg, ...)

## S4 method for signature 'array'
T2.fast(cpmg, cpmg.mask, TE,
   control = minpack.lm::nls.lm.control(maxiter = 150), multicore = FALSE,
   verbose = FALSE)

T2.lm(signal, TE, guess, control = minpack.lm::nls.lm.control())

Arguments

cpmg is a multidimensional array of signal intensities. The last dimension is assumed to be a function of the echo times, while the previous dimensions are assumed to be spatial.

... Additional variables defined by the method.

cpmg.mask is a (logical) multidimensional array that identifies the voxels to be analyzed.

TE is the vector of echo times (in seconds).

control An optional list of control settings for nls.lm. See nls.lm.control for the names of the settable control values and their effect.

multicore is a logical variable (default = FALSE) that allows parallel processing via multicore.

verbose is a logical variable (default = FALSE) that allows text-based feedback during execution of the function.

signal is the vector of signal intensities as a function of echo times.

guess is the vector of initial values for the parameters of interest: \( \rho \) and \( T^2 \).
Value

A list structure is produced with (all or some of the) parameter estimates

- \( \rho \) Scaling factor between signal intensity and T2 (proton density).
- T2 T2 relaxation time.

Author(s)

Brandon Whitcher <bwhitcher@gmail.com>

References


See Also

R1.fast, R10.lm
Index

+Topic **datasets**
buckley, 8

+Topic **misc**
aifParameters, 7
compartmentalModel, 9
expConv, 24
extractAIF, 24
findCenter, 26
R10.lm, 29
rCBV.fast, 32
T2.fast, 34

+Topic **models**
ADC.fast, 3
aif-models, 5
dcemri.bayes, 11
dcemri.lm, 14
dcemri.map, 18
dcemri.spline, 20
doubleAngleMethod, 22
kineticModel, 27

+Topic **package**
dcemriS4-package, 2

ADC.fast, 3
ADC.fast,array-method(ADC.fast), 3
adc.lm(ADC.fast), 3
aif-models, 5
aif.orton.exp(aif-models), 5
aifParameters, 7, 9

breast (buckley), 8
buckley, 8

CA.fast(R10.lm), 29
CA.fast,array-method(R10.lm), 29
CA.fast2(R10.lm), 29
CA.fast2,array-method(R10.lm), 29
compartmentalModel, 7, 9
convFFT, 10, 26, 33
dcemri.bayes, 9, 11, 17, 19, 22, 28
dcemri.bayes,array-method
dcemri.bayes(2cemri.bayes), 11
dcemri.bayes.single(dcemri.bayes), 11
dcemri.lm, 6, 7, 9, 13, 14, 19, 22, 28, 31
dcemri.lm,array-method(dcemri.lm), 14
dcemri.map, 9, 13, 17, 18, 22
dcemri.map,array-method(dcemri.map), 18
dcemri.spline, 13, 17, 20, 28
dcemri.spline,array-method
dcemri.spline(2cemri.spline), 20
dcemri.spline.single(dcemri.spline), 20
dcemriS4(dcemriS4-package), 2
dcemriS4-package, 2
doubleAngleMethod, 22
E10.lm(R10.lm), 29
expConv, 24
extractAIF, 6, 24

fastTemplateMatching, 10, 25, 27
fastTemplateMatching,array-method
(fastTemplateMatching), 25
fft, 10, 26
findCenter, 26, 26

kineticModel, 7, 27

meningioma (buckley), 8
model.orton.exp(aif-models), 5

nls.lm, 4, 6, 17, 30, 31

orton.exp.lm(aif-models), 5

R1.fast, 35
R1.fast(R10.lm), 29
R1.fast,array-method(R10.lm), 29
R1.lm(R10.lm), 29
R10.lm, 29, 35
rCBV(rCBV.fast), 32
rCBV.fast, 32
INDEX

shift3D, 10, 26, 33

T2.fast, 34
T2.fast, anlz-method (T2.fast), 34
T2.fast, array-method (T2.fast), 34
T2.fast, nifti-method (T2.fast), 34
T2.lm (T2.fast), 34