Package 'RfmriVC'

July 2, 2014

Type Package

Title Varying stimulus coefficient fMRI models in R

Version 1.0.4

Date 2013-06-04

Author Ludwig Bothmann, Stefanie Kalus

Maintainer Stefanie Kalus <stefanie.kalus@stat.uni-muenchen.de>

Description A varying coefficient on the stimulus effect is introduced
into the voxelwise fMRI regression model. By this the effects
of the canonical hemodynamic response function are allowed to
vary with another variable, like for example time or a vigilance parameter.

Depends methods, splines, mvtnorm, mgcv, Rniftilib

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2013-06-04 16:02:30

R topics documented:

RfmriVC-package *RfmriVC*

Description

The RfmriVC package provides utilities for fitting a varying stimulus coefficient model within a voxelwise fMRI regression model. In each voxelwise model, the effects of all requested canonical HRF basis functions are allowed to vary depending on the values of a chosen variable. The model fit can be done either with bayesian or classical (penalized least squares by MGCV) techniques.

Details

This package builds upon two S4 classes which control the configuration, running and postprocessing of the varying coefficient fMRI model. An object of class ConfigVC is used to set all configurations needed to run the algorithm (including output of preprocessing steps). An Object of class ResultsVC is returned by a model run (return of doFmriVC()) and can be used to visualize results.

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

Examples

```
## Not run:
library(RfmriVC)

#------- Read data ----------
# Read in fMRI data
y<-nifti.image.read("nifti_fmri.nii")

#Stimulus presentation times
stimTimes<-as.matrix(read.table("stimulusTimes.txt"))

#Confounders
confMat<-as.matrix(read.table("covariates.txt"))


#------- Classical ------------
# Generate configuration object for classical analysis (= default analysis)
configObj<-new("ConfigVC")
configObj<-setConfigVC(configObj,stimTimes=stimTimes,VC=stimTimes,y=y,
                       nStimBF=3,confounder=confMat)
```

```
# Do classical analysis
resObj<-doFmriVC(configObj)

#Look at results of first voxel (1,1,1)
par(mfrow=c(3,1),mar=c(3, 4, 0, 1) + 0.1,oma=c(0, 0, 5, 0))

#Plot stimulus varying coefficient depending on variable VC: beta(VC)
betaHatEst<-betaHat(resObj,voxelIndex3D=c(1,1,1))
matplot(resObj@VC.plot,betaHatEst,type="l",xlab="VC",ylab=expression(beta~(VC)))

#Plot HRF for VC=100
mat<-hrfHat(resObj,VC=100,t=seq(0,30,length.out=1000),voxelIndex3D=c(1,1,1))
matplot(resObj@t.plot,mat,type="l",xlab="t",ylab="hrf(t)")

#Plot HRF for t=5
matVC<-hrfHat(resObj,t=5,VC=seq(7,390,length.out=1000),voxelIndex3D=c(1,1,1))
matplot(resObj@VC.plot,matVC,type="l",xlab="VC",ylab="hrf(VC)")
title("Classical estimation",outer=TRUE,line=2)

#Image plot for HRF surface
windows()
t.grid<-seq(1,15,length.out=50)
VC.grid<-seq(7,670,length.out=50)
matVCt<-hrfHat(resObj,t=t.grid,VC=VC.grid,voxelIndex3D=c(1,1,1))
image(y=VC.grid,x=t.grid,t(matVCt[,,1]),main="Classical estimation")


#------- Bayes ------------
set.seed(1112)
# Generate configuration object for bayesian analysis
# (= non-default analysis -> use extra setter)
configBayObj<-new("ConfigVC")
configBayObj<-setConfigVC(configBayObj,stimTimes=stimTimes,VC=stimTimes,y=y,
                          nStimBF=3,confounder=confMat)
configBayObj<-setConfigVC.Bayes(configBayObj)

#Do analysis
resBayObj<-doFmriVC(configBayObj)

#Look at results
windows()
par(mfrow=c(3,1),mar=c(3, 4, 0, 1) + 0.1,oma=c(0, 0, 5, 0))

#Plot stimulus varying coefficient depending on variable VC: beta(VC)
betaHatEst<-betaHat(resBayObj,voxelIndex3D=c(1,1,1))
beta.hat.plot(resBayObj@VC.plot,betaHatEst)

#Plot HRF for VC=100
mat<-hrfHat(resObj,VC=100,t=seq(0,30,length.out=1000),voxelIndex3D=c(1,1,1))
matplot(resObj@t.plot,mat,type="l",xlab="t",ylab="hrf(t)")

#Plot HRF for t05
matVC<-hrfHat(resObj,t=5,VC=seq(7,390,length.out=1000),voxelIndex3D=c(1,1,1))
```

```
matplot(resObj@VC.plot,matVC,type="l",xlab="VC",ylab="hrf(VC)")
title("Bayesian estimation",outer=TRUE,line=2)

#Image plot for HRF surface
windows()
t.grid<-seq(1,15,length.out=50)
VC.grid<-seq(7,670,length.out=50)
matVCt<-hrfHat(resObj,t=t.grid,VC=VC.grid,voxelIndex3D=c(1,1,1))
image(y=VC.grid,x=t.grid,t(matVCt[,,1]),main="Bayesian estimation")



## End(Not run)
```

`calcDeviceHeight` *Calculates the optimal graphical device height for printing squared voxels*

Description

This helper function calculates for a given width value, e.g., in `pdf(file,width, height, ...)` the according height value, so that voxels appear as squares.

Usage

```
calcDeviceHeight(widthVal, mfrowVal, maiVal, omiVal, niftiDim)
```

Arguments

`widthVal`	Intended `width` value for the graphical device.
`mfrowVal`	The par-parameter `mfrow` intended to be used for plotting.
`maiVal`	The par-parameter `mai` intended to be used for plotting.
`omiVal`	The par-parameter `omi` intended to be used for plotting.
`niftiDim`	A 3D vector with the 3D dimensions of the nifti image object.

Value

`heightVal`	The `height` value for a given `width` value, which leads to square voxels in a graphical device.

Author(s)

Stefanie Kalus <stefanie.kalus <at> stat.uni-muenchen.de>

Examples

```
#Plotting parameters
dimVec<-c(43,40,19)
mfrowVal<-c(4,5)
maiVec<-c(0,0, 0,0)+0.015
omiVec<-c(0,0,0.65,0)+0.15 #Outer title margin

#Calculate appropriate width and height, so that image ratios are preserved
widthVal<-7
heightVal<-calcDeviceHeight(widthVal,mfrowVal,maiVec,omiVec,dimVec)
heightVal
```

ConfigVC-class *Class* "ConfigVC"

Description

This is a class for configurating the RfmriVC-algorithm. The slots of the "ConfigVC"-class contain all arguments and helper variables which are passed to the RfmriVC-algorithm. Two setter functions are provided to structure the setting of slots resp. arguments with regard to content. If kept unspecified in calls of the setters, slots are assigned default values.

Objects from the Class

Objects can be created by calls of the form new("ConfigVC", ...). To initializes an object use setter functions setConfigVC and afterwards, if a bayesian analysis is requested setConfigVC.Bayes.

Slots

tr: Repetition time of fMRI scans in seconds.

stimTimes: Numeric vector with stimulus presentation times of one stimulus type which should have a varying coefficient.

nStimBF: Number of canonical HRF basis functions: 1 = just canonical HRF, 2 = plus time derivative, 3 = plus dispersion derivative.

nVCknots: Number of knots for the varying coefficient (VC) spline.

degreeVCspline: Degree of the VC spline.

VC: Numeric vector with variable values the VC depends on.

confounder: Matrix with confounder values.

freqHPF: Frequency of high pass filter (HPF), e.g. 128s.

y: 4d fMRI dataset in form of an object of class "nifti".

mask: 3D analysing mask in form of an object of class "nifti".

mask2totalIndex: Matrix with mapping from mask index to total index, i.e. information about the i-th selected voxel is contained in row i, which contains the 3D index of the corresponding voxel.

`total2maskIndex`: Object of class `"nifti"`, which contains at i,j,k the index of the voxel within the set of selected voxels if selected by mask or zero otherwise.

`grandMeanScaling`: Indicator if a grand mean scaling should be applied to the fMRI data.

`U`: Design matrix part with nuisance covariables (HPF baseline and confounders).

`X`: Design matrix part with

`penMat`: Penalty matrix for one VC spline.

`method`: The method to be applied for estimation and inference. Either "classical" (penalized least squares) or "bayes".

`methodParams`: If `method`=="bayes", a list with configuration settings for the bayesian method.

Methods

The following methods operate on objects of class `ConfigVC`. Details of these methods are described on separate manual pages.

doFmriVC `signature(object = "ConfigVC")`: ...

doFmriVC.Bayes `signature(object = "ConfigVC")`: ...

doFmriVC.Classical `signature(object = "ConfigVC")`: ...

setConfigVC.Bayes `signature(object = "ConfigVC")`: ...

setConfigVC.Classical `signature(object = "ConfigVC")`: ...

setConfigVC `signature(object = "ConfigVC")`: ...

show `signature(object = "ConfigVC")`: ...

saveConfig `signature(object = "ConfigVC")`: ...

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

See Also

Setter functions for assigning slot values (constructors):
`setConfigVC`, `setConfigVC.Bayes`, `setConfigVC.Classical`

Running the estimation procedure:
`doFmriVC`, `doFmriVC.Classical`, `doFmriVC.Bayes`

Saving and loading an external representation of a ConfigVC-object:
`saveConfig`, `loadConfig`

ConfigVC-doFmriVC *Function to fit a varying stimulus coefficient within an fMRI regression model.*

Description

Depending on the configuration settings in a ConfigVC object, these functions start the estimation process: doFmriVC(object) either starts the bayesian or classical procedure depending on the value of slot method. Alternatively, the classical procedure can be started directly by calling doFmriVC.Classical(object) and the bayesian procedure can be strated directly by calling doFmriVC.Bayes(object).

Usage

```
## S4 method for signature 'ConfigVC'
doFmriVC(object)

## S4 method for signature 'ConfigVC'
doFmriVC.Classical(object)

## S4 method for signature 'ConfigVC'
doFmriVC.Bayes(object)
```

Arguments

object	Object of class ConfigVC.

Value

resultsObj

resultsObj:	Object of class ResultsVC.

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

See Also

ConfigVC, RfmriVC

Examples

```
## Not run:
See package RfmriVC documentation: package?RfmriVC

## End(Not run)
```

ConfigVC-setters *ConfigVC-Class: Constructor resp. setter functions*

Description

These functions serve as constructors for ConfigVC objects, i.e. these should be used to assign values to object slots.

Usage

```
## S4 method for signature 'ConfigVC'
setConfigVC(object,tr=2,stimTimes,nStimBF=1,VC,nVCknots=10,
degreeVCspline=2,confounder=NULL,freqHPF=128,y,mask=NULL,
    grandMeanScaling=TRUE)

## S4 method for signature 'ConfigVC'
setConfigVC.Classical(object)

## S4 method for signature 'ConfigVC'
setConfigVC.Bayes(object,chainSize=1100,burnin=100,step=5,sigma2start=1,
sigma2IG=list(a=0.001,b=0.001),VCcontrol=NULL,diagSample=NULL)
```

Arguments

object	Object of class ConfigVC.
tr	Repetition time of fMRI scans in seconds. Default: tr=2
stimTimes	Numeric vector with stimulus presentation times of one stimulus type which should have a varying coefficient.
nStimBF	Number of canonical HRF basis functions: 1 = just canonical HRF, 2 = plus time derivative, 3 = plus dispersion derivative. Default: nStimBF=1
VC	Numeric vector with variable values the VC depends on.
nVCknots	Number of knots for the varying coefficient (VC) spline. Default: nVCknots=10
degreeVCspline	Degree of the VC spline. Default: degreeVCspline=2
confounder	Matrix with confounder values.
freqHPF	Frequency of high pass filter (HPF). Default: freqHPF=128
y	4d fMRI dataset in form of an object of class "nifti".
mask	3D analysing mask in form of an object of class "nifti". Per default a threshold mask is clculated.

`grandMeanScaling`
: Indicator if a grand mean scaling should be applied to the fMRI data.

`chainSize`
: Total number of MCMC iterations to be conducted. Default: `chainsize=1100`

`burnin`
: Number of iterations after which convergency of Markov chains to their equilibrium distribution is assumed. Default: `burnin=100`

`step`
: Thinning parameter: After the burnin phase one out of `step`-th samples of the Markov chains is used for estimation. Default: `step=5`

`sigma2start`
: Starting parameter for each voxels residual variance 'sigma2'.

`sigma2IG`
: A list with two elements containing the inverse gamma shape and scale parameter of the 'sigma2' prior distribution.

`VCcontrol`
: A list with with `nStimBF` elements. Each element is a list with configurations for one HRF basis function. See details.

`diagSample`
: Vector with 1D indices of voxels for which all diagnostics should be safed.

Details

For each HRF basis function, which is requested to be contained in the model, the `VCcontrol`-list contains a list with the following default: list(gammaStart=matrix(0,nrow=p,ncol=1),a=0.001,b=0.001). If non-default values should be used, the whole `VCcontrol` statement must be given, e.g. for a model with canonical HRF and its time derivative (`nStimBF=2`):

`VCcontrol`=list(list(gammaStart=matrix(gammaLSest1,nrow=p,ncol=1),a=0.05,b=0.001),

list(gammaStart=matrix(gammaLSest2,nrow=p,ncol=1),a=0.1,b=0.1))

where parameter p is the number of VC coefficients of each HRF (p<-nVCknots+degreeVCspline-1) and gammaLSest1 and gammaLSest2 are for example the estimates from a model with constant stimulus effects.

Value

`object`
: Object of class `ConfigVC`.

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

See Also

`ConfigVC`, RfmriVC

Examples

```
## Not run:
See package RfmriVC documentation: package?RfmriVC

## End(Not run)
```

---

plotNifti *Function to plot brain image slices*

Description

A function to plot the layers of one 3D nifti image stored in a nifti object of library Rniftilib.

Usage

```
plotNifti(nifti.img, zRange=NULL, zVec=NULL,  mfrowVal, plotTitle=TRUE,
          pdfoutput=FALSE, pdfName=NULL, scale=1,...)
```

Arguments

nifti.img	A nifti object of library Rnfitilib which stores the 3D (4D) nifti image.
zRange	An optional two-dimensional integer vector with a start and end layer to be plotted.
zVec	An optional integer vector with all layer numbers to be plotted.
mfrowVal	An optional 2D vector in form of c(nRow,nCol). This is for par(mfrow=mfrowVal).
plotTitle	A bool value stating whether a title should be plotted above image plots.
pdfoutput	A bool value stating whether a pdf should be generated.
pdfName	An optional string with a name for the pdf file.
scale	An optional scale value which can be used to scale the values of the plotted image.
...	Optional arguments which can be passed to par(...)

Note

If a 4D image is given the nifti image only the first 3D nifti image is plotted.

Author(s)

Stefanie Kalus <stefanie.kalus <at> stat.uni-muenchen.de>

Examples

```
## Not run:
require("Rniftilib",character.only=TRUE)
activMap<-nifti.image.read("activMap.nii")
plotNifti(activMap,pdfoutput=TRUE)

## End(Not run)
```

ResultsVC-betaHat *Calculation of the stimulus varying coefficient courses*

Description

Postprocessing routine: Functions to calculate the course of the stimulus varying coefficient of a selected HRF basis function with corresponding significance/credibility intervals

Usage

```
## S4 method for signature 'ResultsVC'
betaHat(object,voxelIndex3D,VC.plot=object@VC.plot,which.beta=1,alpha=object@alpha)

## S4 method for signature 'ResultsVC'
betaHatNifti(object,VC.plot=object@VC.plot,which.beta=1)
```

Arguments

object	An object of class ResultsVC.
voxelIndex3D	A 3D vector with the voxel index for which betaHat estimates should be calculated.
VC.plot	Vector of VC variable values for which betaHat estimates should be calculated.
which.beta	Number of requested basis function (1 = canonical HRF, 2 = time derivative, 3 = dispersion derivative).
alpha	Significance resp. credibility level. Default: 0.05.

Value

betaHat returns

betaHat	Matrix of size length(VC.plot) x 3 with betaHat-estimate (1st column), lower and upper confidence interval limits (2nd + 3rd column) for a selected voxel.

betaHatNifti returns

betaHatTotal	4D nifti image with betaHat vector estimate for each voxel.

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

See Also

ResultsVC, RfmriVC

Examples

```
## Not run:
See package RfmriVC documentation: package?RfmriVC

## End(Not run)
```

ResultsVC-class *Class* "ResultsVC"

Description

This is a class which stores results from a varying stimulus coefficient fMRI regression (doFmriVC()) and further objects needed for postprocessing the results.

Objects from the Class

Objects are generated by a call to doFmriVC(). If necessary they can be created by calls of the form new("ResultsVC", ...) or modified by calls to resetResultsVC().

Slots

VC: Numeric vector with variable values the VC depends on.

VC.plot: Vector of VC variable values for which betaHat estimates should be calculated.

nVCknots: Number of knots for the varying coefficient (VC) spline.

degreeVCspline: Degree of the VC spline.

stimTimes: Numeric vector with stimulus presentation times of one stimulus type which should have a varying coefficient.

nStimBF: Number of canonical HRF basis functions: 1 = just canonical HRF, 2 = plus time derivative, 3 = plus dispersion derivative.

U: Design matrix part with nuisance covariables (HPF baseline and confounders).

X: Design matrix part with

penMat: Penalty matrix for one VC spline.

total2maskIndex: Object of class "nifti", which contains at i,j,k the index of the voxel within the set of selected voxels if selected by mask or zero otherwise.

t.plot: Vector of (peristimulus) time values for which hrf estimates should be calculated.

alpha Significance resp. credibility level. Default: 0.05.

method: The method which has been applied for estimation and inference. Either "classical" (penalized least squares) or "bayes".

methodResults: A list with estimation results.

Methods

The following methods are dsigned as member functions of this class. Documentation of these functions can be found on separate manual pages.

betaHat `signature(object = "ResultsVC")`: ...

betaHatNifti `signature(object = "ResultsVC")`: ...

zeroCoverageNifti `signature(object = "ResultsVC")`: ...

hrfHat `signature(object = "ResultsVC")`: ...

hrfHatPoint `signature(object = "ResultsVC")`: ...

resetResultsVC `signature(object = "ResultsVC")`: ...

setResultsVC `signature(object = "ResultsVC")`: ...

show `signature(object = "ResultsVC")`: ...

saveResults `signature(object = "ResultsVC")`: ...

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

See Also

`doFmriVC`, `hrfHat`, `betaHat` RfmriVC

Saving and loading an external representation of a ConfigVC-object:
`saveResults`, `loadResults`

Examples

```
showClass("ResultsVC")
```

`ResultsVC-hrfHat` *Calculation of HRF estimates*

Description

Postprocessing routine: Functions to calculate the HRF estimates with corresponding significance / credibility intervals.

`hrfHatPoint()` calculates the hrf estimate for one pair of t and VC value. `hrfHat()` calculates the hrf estimate for scalar or vector valued t and VC settings.

Usage

```
## S4 method for signature 'ResultsVC'
hrfHatPoint(object,VCVal=median(object@VC),tVal,voxelIndex3D,alpha=object@alpha)

## S4 method for signature 'ResultsVC'
hrfHat(object,VC=median(object@VC),t=object@t.plot,voxelIndex3D,alpha=object@alpha)
```

Arguments

object	An object of class ResultsVC.
VCVal	Varying coefficient variable value for which hrf(VC,t) should be calculated.
VC	Either a value or a vector of the VC variable for which hrf(VC,t) should be calculated.
tVal	Peristimulus time value for which hrf(VC,t) should be calculated.
t	Either a value or a vector of (peristimulus) time for which hrf(VC,t) should be calculated.
voxelIndex3D	A 3D vector with the voxel index for which betaHat estimates should be calculated.
alpha	Significance resp. credibility level. Default: 0.05.

Value

hrfHatPoint returns

hrfHatPoint	Vector with hrfHat-estimate (1st element), lower and upper confidence interval limits (2nd + 3rd elements) for the requested t and VC value for a selected voxel.

hrfHat returns

hrfHat	Vector, matrix or array with hrfHat-estimate (1st element/column/layer), lower and upper confidence interval limits (2nd + 3rd elements/columns/layers) for a selected voxel.

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

See Also

ResultsVC, RfmriVC

Examples

```
## Not run:
See package RfmriVC documentation: package?RfmriVC

## End(Not run)
```

ResultsVC-setters *Setter functions for class* ResultsVC

Description

These functions serve as constructors for ResultsVC objects, i.e. these should be used to assign values to object slots. setResultsVC() is primarily used internally and is called by doFmriVC(). resetResultsVC() can be used to modify slots which are of interest for the user.

Usage

```
## S4 method for signature 'ResultsVC'
setResultsVC(object,configVCobj,method,methodResults)

## S4 method for signature 'ResultsVC'
resetResultsVC(object, VC.plot=seq(min(configVCobj@VC),max(configVCobj@VC),
length.out=1000), t.plot=seq(0,30,length.out=1000), alpha=0.05)
```

Arguments

object	An object of class ResultsVC.
configVCobj	An object of class ConfigVC.
method	The method which has been applied for estimation and inference. Either "classical" (penalized least squares) or "bayes".
methodResults	A list with estimation results.
VC.plot	Vector of VC variable values for which betaHat estimates should be calculated.
t.plot	Vector of (peristimulus) time values for which hrf estimates should be calculated.
alpha	Significance resp. credibility level. Default: 0.05.

Value

object	An object of class ResultsVC.

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

See Also

ResultsVC, RfmriVC

RfmriVC-save *Functions to save and load* ConfigVC *and* RfmriVC *objects from the RfmriVC package*

Description

saveConfig writes an external representation of a ConfigVC object to the specified file. loadConfig reloads datasets written with the function saveConfig. saveResults writes an external representation of a ResultsVC object to the specified file. loadResults reloads datasets written with the function saveResults.

Usage

```
## S4 method for signature 'ConfigVC'
saveConfig(object,path)

loadConfig(path)

## S4 method for signature 'ResultsVC'
saveResults(object,path)

loadResults(path)
```

Arguments

object	Object of class ConfigVC or ResultsVC.
path	Path to folder, where objects should be saved resp. was saved.

Details

Extra functions are necessary because otherwise the nifti-members are not written to file.

Value

object:	Object of either class ConfigVC or ResultsVC.

Author(s)

Ludwig Bothmann and Stefanie Kalus

References

Bothmann, L. (2012). Statistische Modellierung von EEG-abhaengigen Stimuluseffekten in der fMRT-Analyse. Diploma thesis. LMU Munich: Germany.

See Also

save, load, ConfigVC, ResultsVC, RfmriVC

Examples

```
## Not run:
See package RfmriVC documentation: package?RfmriVC

## End(Not run)
```

Index