Package ‘vows’

March 22, 2013

Type Package

Title Voxelwise semiparametrics

Version 0.2-1

Date 2013-03-22

Author Philip Reiss, Yin-Hsiu Chen, Lei Huang, and Lan Huo

Maintainer Lan Huo <lan.huo@nyumc.org>

Depends R (>= 2.9.0), fda, gamm4, RLRsim, Rniftilib, rpanel (>= 1.1-1), shape, stringr, tcltk, tkrplot

Description Parametric and semiparametric inference for massively parallel models, i.e., a large number of models with common design matrix, as often occurs with brain imaging data.

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2013-03-22 20:14:38

R topics documented:

vows-package ......................................................... 2
F.mp .............................................................. 3
Fdr.rlrt .......................................................... 4
funkmeans ......................................................... 5
funkmeans4d ....................................................... 6
funkpanel .......................................................... 7
lm.mp .............................................................. 8
lm4d ............................................................... 9
nii2R ............................................................. 10
Voxelwise semiparametrics

Description

This package efficiently performs inference on a large set of parametric or semiparametric regressions that are "parallel" in the sense that they have a common design matrix. The functions are inspired by neuroimaging applications, where the parallel models pertain to a grid of brain locations known as voxels.

Details

Functions ending in ".mp" ("massively parallel") are designed for responses in the form of a (wide) matrix; functions ending in "4d" take four-dimensional response data (e.g., a set of images) and convert it to matrix form so that the corresponding ".mp" function can be applied. Examples include lm.mp and lm4d for ordinary linear models, rlrt.mp and rlrt4d for restricted likelihood ratio tests (RLRTs) of a parametric null hypothesis vs. a smooth alternative, and semipar.mp and semipar4d for smoothing (see Reiss et al., 2013). Functions for interactive visualization (rlrtpanel and funkpanel) are also provided.

Author(s)

Philip Reiss <phil.reiss@nyumc.org>, Yin-Hsiu Chen <enjoychen0701@gmail.com>, Lei Huang <huangracer@gmail.com>, and Lan Huo <lan.huo@nyumc.org>

Maintainer: Lan Huo <lan.huo@nyumc.org>
References


---

**F.mp**

*F*-tests for massively parallel linear models

---

**Description**

Performs F-tests for removing one or more terms from each of a large number of models with common design matrix.

**Usage**

```r
F.mp(formula, which)
```

**Arguments**

- `formula`: a formula such as "Y ~ X", where Y is an n × V response matrix and X is an n × p design matrix common to all V models.
- `which`: number or vector indicating which column(s) of the model matrix are to be tested for removal from the model.

**Value**

- `F`: F-statistics for each of the models.
- `df1`: numerator degrees of freedom.
- `df2`: denominator degrees of freedom.
- `pvalue`: upper-tailed p-value.
- `X`: design matrix.

**Author(s)**

Philip Reiss <phil.reiss@nyumc.org> and Lei Huang <huangracer@gmail.com>

**See Also**

- `lm.mp`, `permF.mp`

**Examples**

```r
Y = matrix(rnorm(6000), nrow=20)
X = rnorm(20)
t2 = F.mp(Y~X, which=2)
```
Description

Given a set of RLRT results and a threshold, this function outputs an estimate of the FDR (in the empirical Bayes sense of Efron, 2010) when the given threshold is used to determine which null hypotheses to reject.

Usage

Fdr.rlrt(rlrt.obj, threshold)

Arguments

| rlrt.obj | an RLRT object obtained from rlrt.mp or rlrt4d. |
| threshold | threshold at which the null hypothesis is rejected. |

Value

A list with elements

- **MoM** FDR based on method of moments estimator of RLRT parameters (Greven et al., 2008).
- **ML** FDR based on maximum likelihood estimation of RLRT parameters, as described in Greven et al. (2008).

Author(s)

Philip Reiss <phil.reiss@nyumc.org>

References


See Also

rlrt.mp, rlrt4d

Examples

# See example for rlrt.mp
Description

This function performs k-means clustering for curve estimates corresponding to each of a 3D grid of points. For example, when scatterplot smoothing is performed at each of a grid of brain voxels as in Reiss et al. (2013), this function can be used to cluster the obtained smooths.

Usage

funkmeans(fdobj, deriv = 1, lambda = 0, ncomp, centers, nstart = 10)

Arguments

fdobj a functional data object, of class "fd", defining the set of curves being clustered.
deriv which derivative of the curves should be clustered. If 0, the curves themselves are clustered; if 1 (the default), their first derivatives are clustered, a natural way to assign curves of similar shape to the same cluster.
lambda smoothing parameter for functional PCA as implemented by pca.fd.
ncomp number of functional principal components.
centers number of clusters.
nstart number of randomly chosen sets of initial centers used by the kmeans function.

Details

The functional clustering algorithm consists of performing (i) functional principal component analysis of the curve estimates or their derivatives, followed by (ii) k-means clustering of the functional PC scores (Tarpey and Kinateder, 2003).

Value

An object of class "funkmeans", which is a list with elements:

cluster, centers, withins, tots, tot.withins, betweenness, size

see kmeans.
basis,coef basis object and coefficient matrix defining the functional data object (see fd) for the curves that are clustered.
fpca functional principal components object, output by pca.fd.
R2 proportion of variance explained by the k clusters.

Author(s)

Philip Reiss <phil.reiss@nyumc.org>, Lei Huang <huangracer@gmail.com> and Lan Huo <lan.huo@nyumc.org>
References


See Also

*funkmeans4d, funkpanel*

Examples

# See example for funkpanel

---

**funkmeans4d**

Functional k-means clustering for parallel smooths for 4-dimensional data

Description

This is a wrapper function for *funkmeans* to handle 3D image responses.

Usage

`funkmeans4d(fdobj, arr4d, ...)`

Arguments

- `fdobj`: a functional data object, of class "fd", defining the set of curves being clustered.
- `arr4d`: a 4-dimensional array containing the raw data that were smoothed at each point. The first 3 dimensions refer to x, y, and z coordinates and the last dimension corresponds to different images.
- `...`: other arguments, passed to *funkmeans*.

Value

An object of class "funkmeans4d", which is also of class "funkmeans" but has the additional component *arr.cluster*: an array, of dimension `dim(arr4d)[1:3]`, giving the cluster memberships.

Author(s)

Philip Reiss <phil.reiss@nyumc.org>, Lei Huang <huangracer@gmail.com> and Lan Huo <lan.huo@nyumc.org>
**funkpanel**

See Also

funkmeans, funkpanel

Examples

# See example for funkpanel

```r
funkpanel
```

**Description**

This function uses interactive graphics tools, provided by the **rpanel** package (Bowman et al., 2007), to visualize the results of functional k-means clustering as implemented by **funkmeans**.

**Usage**

```r
funkpanel(fkmobj, semiobj, arr4d, predictor, titl = "", xlab = "", ylab = "", ncluster = nrow(fkmobj$centers), slice = dim(fkmobj$arr.cluster)[3]%/%2, ylim.scatter = NULL, deriv.legend = 0, ylim.legend = NULL, scattermain = NULL, colvec = NULL)
```

**Arguments**

- `fkmobj` a functional k-means clustering object obtained from **funkmeans4d**.
- `semiobj` the massively parallel smoothing object on which the clustering was based; ordinarily produced by **semipar.mp** or **semipar4d**.
- `arr4d` a 4-dimensional array containing the raw data that were smoothed at each point. The first 3 dimensions refer to x, y, and z coordinates and the last dimension corresponds to different images.
- `predictor` a vector or matrix of covariates.
- `titl` title of the panel.
- `xlab`, `ylab` x- and y-axis labels.
- `ncluster` number of clusters to display. By default, all are displayed.
- `slice` index of the slice to be shown initially in the panel.
- `ylim.scatter` the y limits of the voxelwise scatterplots.
- `deriv.legend` which derivative to plot in the "legend"; see Details. By default, the curves themselves are used.
- `ylim.legend` the y limits used in the "legend"; see Details.
- `scattermain` title for the scatter plots.
- `colvec` a vector of colors for the clusters. By default, this is set to the first `ncluster` elements of `c("dodgerblue", "green", "red", "orange", "yellow", "orchid", "brown", "grey", "purple")` if `ncluster < 9`. 

**Examples**

```r
# See example for funkpanel
```
Details
The clusters defined by the input object are shown for a cross-section (slice), and a slider allows viewing of different slices. Clicking on a voxel produces a scatterplot of the data at that voxel, along with the fitted scatterplot. The "legend", produced by `plot.funkmeans`, consists of plots of 30 randomly selected curves, along with the cluster center, from each cluster.

Author(s)
Lei Huang <huangracer@gmail.com>, Yin-Hsiu Chen <enjoychen0701@gmail.com>, and Lan Huo <lan.huo@nyumc.org>

References

See Also
funkmeans, funkmeans4d, plot.funkmeans

Examples
```r
data(test)
d4 = test$d4
x = test$x
semi.obj = semipar4d(d4, ~sf(x), -5:5, data.frame(x = x))
temp = semi.obj$list.all[[1]]
fdobj = fd(coef = semi.obj$coef[(temp$start):(temp$end), ], basisobj=temp$basis)
fkmobj = funkmeans4d(fdobj, d4, ncomp=6, centers=3)
funkpanel(fkmobj, semi.obj, d4, x)
```

lm.mp
Massively parallel linear regression models

Description
Efficiently fits $V$ linear models with a common design matrix, where $V$ may be very large, e.g., the number of voxels in a brain imaging application.

Usage
```r
lm.mp(Y, formula, store.fitted = FALSE)
```

Arguments
Y $n \times V$ outcome matrix.
formula a formula object such as "~ x1 + x2".
store.fitted logical: Should the fitted values be stored? For large $V$, setting this to TRUE may cause memory problems.
lm4d

Value

- coef: $p \times V$ matrix of coefficient estimates.
- sigma2: $V$-dimensional vector of error variance estimates.
- se.coef: $p \times V$ matrix of coefficient standard error estimates.
- X: $n \times p$ common design matrix.
- fitted: $n \times V$ matrix of fitted values.

Author(s)

Philip Reiss <phil.reiss@nyumc.org>, Lei Huang <huangracer@gmail.com>, and Yin-Hsiu Chen <enjoychen0701@gmail.com>

See Also

lm4d, summary.lm.mp

Examples

# Please see example for lm4d

Description

This is a wrapper function for lm.mp to handle 3D image responses.

Usage

lm4d(arr4d, formula, store.fitted = FALSE)

Arguments

- arr4d: a 4-dimensional response array, where the first 3 dimensions refer to spatial coordinates and the last dimension corresponds to different images.
- formula, store.fitted

Value

An object of class "lm.mp", with two changes. (1) If store.fitted = TRUE, the fitted values are given as a 4-dimensional array. (2) A call component is included.

Author(s)

Lei Huang <huangracer@gmail.com>, Yin-Hsiu Chen <enjoychen0701@gmail.com>, and Philip Reiss <phil.reiss@nyumc.org>
See Also

lm.mp

Examples

data(test)
d4 = test$d4
x = test$x
lmobj = lm4d(d4, ~x)

# Convert d4 to a matrix, and confirm that lm.mp() gives the same results as lm4d()
d4.2 = d4
dim(d4.2) = c(prod(dim(d4)[1:3]), dim(d4)[4])
Y = t(d4.2)
lmobj2 = lm.mp(Y, ~x)
all.equal(lmobj$coef, lmobj2$coef)

nii2R

NIfTI-to-R conversion

Description

Reads in a NIfTI (.nii) file and puts the data in a 4-dimensional array.

Usage

nii2R(niifilename, which.vols = NULL, savename = NULL, remove.zero = TRUE,
      maskname = NULL, ind = NULL, ind.auto = TRUE, coord = NULL)

Arguments

niifilename the path for the .nii file.
which.vols which volumes (images) to include. In terms of the 4D array, this refers to sub-setting in the fourth dimension. If NULL (the default), all volumes are included.
savename if non-NULL, the name of the .RData file to which the 4D array will be saved.
remove.zero optional when maskname is not provided. If TRUE, a binary array indicating the voxels with nonzero measures based on the first three dimension of the nii file will be provided. If FALSE, a 3D array with TRUE everywhere will be provided.
maskname name of a .nii file providing a "mask", a 3D binary array indicating which voxels to include.
ind, ind.auto ind is an optional list saying which indices (which slices of the image) to include in each of the three dimensions. If NULL, this will be all slices with nonzero data if ind.auto = TRUE, and all slices otherwise.
coord coordinates of the first three dimensions of the 4D array created.
permF.mp

Value

a 4-dimensional array.

Author(s)

Lei Huang <huangracer@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>

See Also

R2nii

Description

Performs permutation F-tests for parallel linear models with a common design matrix. Currently restricted to testing with the intercept-only model as the null hypothesis. The permutation method controls the familywise error rate (FWER) at a desired level; see Details.

Usage

permF.mp(formula, nperm = 499, alpha = 0.05, report.every = 50)

Arguments

formula a formula such as "Y ~ X", where Y is an n × V response matrix and X is an n × p design matrix common to all V models.
nperm number of permutations.
alpha level at which to control the FWER.
report.every parameter controlling how often to report the number of permutations performed; by default, every 50.

Details

The observed F-statistics are referred to a permutation distribution of the maximum F-statistic over all V tests. This is a standard approach to FWER control in neuroimaging (Nichols and Holmes, 2001).

Value

maxF.perm maximal F-statistics obtained from each of the permuted data sets.
F.obs the observed F-statistics.
threshold critical value obtained from the permutations.
pvalue adjusted (familywise error rate-controlling) p-values.
Author(s)
Philip Reiss <phil.reiss@nyumc.org> and Lei Huang <huangracer@gmail.com>

References

See Also
F.mp

Examples
Y = matrix(rnorm(6000), nrow=20)
X = rnorm(20)
t3 = permF.mp(Y~X)

plot.funkmeans

Plotting of k-means clustering results for massively parallel smooths

Description
Visualization of functional k-means clustering as implemented by funkmeans.

Usage
## S3 method for class 'funkmeans'
plot(x, fdobj, deriv = 0, ylim = NULL, ncluster = nrow(x$centers), mfrow = NULL, colvec = NULL, ...)

Arguments
x a functional k-means clustering object obtained from funkmeans.
fdobj a functional data object, of class "fd", defining the set of curves being clustered. See the example.
deriv which derivative to display in the plots, which show 30 randomly selected curves, along with the cluster center, from each cluster. By default, the "0th derivative" is used (i.e., the curves themselves).
ylim the y limits for the plots.
ncluster number of clusters to display. By default, all are displayed.
mfrow a vector of length 2 giving the numbers of rows and columns for the array of plots. By default, the number of rows will exceed the number of columns by 0 or 1, depending on ncluster.
colvec a vector of colors for the clusters. By default, this is set to the first ncluster elements of c("dodgerblue", "green", "red", "orange", "yellow", "orchid", "brown", "grey", "purple"), if ncluster <= 9.
... arguments passed to plot.
Author(s)
Yin-Hsiu Chen <enjoychen0701@gmail.com>, Philip Reiss <phil.reiss@nyumc.org> and Lan Huo <lan.huo@nyumc.org>

See Also
funkmeans, funkpanel

Examples
data(test)
d4 = test$d4
x = test$x
semi.obj = semipar4d(d4, formula = ~sf(x), data = data.frame(x = x), lsp=-5:5)
fdobj = fd(coef = semi.obj$coef[-1, ], basis = semi.obj$list.all[[1]]$basis)
fkmobj = funkmeans(fdobj, ncomp = 8, centers = 6)
plot(fkmobj, fdobj)

plot.rlrt4d

Display cross-sections of voxelwise RLRT results

Description
Plots slices of the 3D array representing a set of voxelwise RLRT results.

Usage
## S3 method for class 'rlrt4d'
plot(x, array4d, disp = c("stat", "p", "fdr", "pwdf"), titl = NULL,
slices = NULL, colbar = TRUE, col.image = femmecol(1/0/1:1),
neglog = FALSE, threshold = NULL, mar = c(2, 2, 2, 2),
digit = 2, nrow = NULL, ...)

Arguments
x a voxelwise RLRT object as produced by rlrt4d.
array4d the 4D array on which the voxelwise RLRT was performed.
disp, titl, neglog, threshold see rlrtpanel.
slices indices of the slice(s) to be displayed.
colbar logical: Should a color bar be included?
col.image color scheme for the color bar, as generated by rainbow, heat.colors, etc.
mar A numerical vector of the form c(bottom, left, top, right) specifying the number of lines of margin on the four sides of the plot.
digit number of significant digits in labels.
nrow number of rows on the plot.
... arguments passed to plot.
**plot.semipar.mp**

**Author(s)**
Lei Huang <huangracer@gmail.com>, Philip Reiss <phil.reiss@nyumc.org> and Lan Huo <lan.huo@nyumc.org>

**See Also**
rlrt4d, rlrtpanel

**Examples**

# Please see the example for rlrt4d

---

**plot.semipar.mp**  
*Plot massively parallel semiparametric models*

**Description**
Given a massively parallel smoothing object produced by `semipar.mp`, the function plots the fitted smooth(s) for a given point (e.g., at a given voxel).

**Usage**

```r
## S3 method for class 'semipar.mp'
plot(x, Y, arr.ind = NULL, which.vox = NULL, which.smooth = NULL,
    coverage = NULL, length.new = 1, ylim = NULL, ylab = NULL, ...)
```

**Arguments**

- `x` an object of class "semipar.mp".
- `Y` an $n \times V$ outcome matrix.
- `arr.ind` a 3-element vector specifying the element of the 3-dimensional array of locations (e.g., voxels) for which plotting is desired. If NULL, `which.vox` must be specified.
- `which.vox` the index of the voxel to be plotted. If NULL, `arr.ind` must be specified.
- `which.smooth` the index of the smooth term of which the confidence interval plot is to be displayed. The default value is NULL which refers to displaying the plots for all the smooth terms in the model.
- `coverage` the confidence level of the pointwise confidence intervals in the plot.
- `length.new` length of the vector of ordered variables with which to predict.
- `ylim, ylab, ...` arguments to be passed to `plot`.

**Author(s)**
Yin-Hsiu Chen <enjoychen0701@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>
Examples

n<-32
Ys <- matrix(0, n, 5)
for(i in 1:n) Ys[i,]<--2:2+rnorm(5, i^2, i^0.5)+sin(i)
x1 <- rnorm(n,0,5)
x2 <- 1:n+runif(n, 1, 20)
semipar.obj <- semipar.mp(~x1+sf(x2,k=1),Y=Ys,lsp=seq(5,5,,3))
plot(semipar.obj, Y=Ys, which.vox=2)

qplsc.mp

Quadratically penalized least squares with constraints

Description

Fits a possibly very large number of models, with common design matrix, by quadratically penalized least squares, with identifiability constraints imposed. This function serves as the fitting engine for semipar.mp.

Usage

qplsc.mp(Y, modmat, penmat, constr.list = NULL, lsp, nulldim = NULL,
store.reml = FALSE, store.fitted = FALSE)

Arguments

Y an n × V response matrix (V refers to number of models fitted in parallel, e.g., voxels in neuroimaging applications).
modmat model matrix, e.g., a matrix of B-spline basis functions.
penmat penalty matrix.
constr.list a list of length equal to number of constraints to be imposed, containing information for reparametization to an unconstrained optimization. Attribute 'C' is the constraint matrix, and 'start' and 'end' refer to the corresponding column positions of the model matrix.
lsp vector of candidate tuning parameters (log(λ)).
nulldim null space dimension, ordinarily equal to the order of the derivative penalty.
store.reml logical: should the pointwise REML criterion at each grid point be included in the output? FALSE by default, as this output can be very large.
store.fitted logical: should the fitted values be included in the output? FALSE by default.

Value

An object of class "qplsc.mp", which is a list with elements:

fitted fitted value matrix, if store.fitted = TRUE.
edf matrix giving the effective degrees of freedom per parameter, as in Wood (2004), for each model.
pwdf vector of point-wise degrees of freedom, equal to the column sums of edf.
pwlsp vector of point-wise log smoothing parameters.
coef matrix of coefficients.
reml matrix giving the point-wise REML criterion at each grid point, if store.reml = TRUE.
modmat model matrix.
penmat penalty matrix.
RinvU \( R^{-1}U \), as in Reiss et al. (2013); this and tau are used for plotting.
tau singular values of \( R^{-T}PR^{-1} \), as in Reiss et al. (2013).
sigma2 vector of variance estimates.
ttu matrix for transformation to an unconstrained problem.

Author(s)
Lei Huang <huangracer@gmail.com>, Yin-Hsiu Chen <enjoychen0701@gmail.com>, and Philip Reiss <phil.reiss@nyumc.org>

References


Examples
```r
## see semipar.mp
```

R2nii

**Save data to a NIfTI file**

Description
This function can be used to output the results of voxelwise RLRT or smoothing.

Usage
```r
R2nii(arr, name.nii)
```

Arguments
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arr</td>
<td>a 3D or 4D array containing data to be saved.</td>
</tr>
<tr>
<td>name.nii</td>
<td>filename, excluding the .nii extension.</td>
</tr>
</tbody>
</table>
Value

None; a NIfTI file is created.

Author(s)

Lei Huang <huangracer@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>

See Also

nii2R

---

rlrt.mp  Massively parallel restricted likelihood ratio tests

Description

Conducts a possibly very large number of restricted likelihood ratio tests (Crainiceanu and Ruppert, 2004), with common design matrix, for a polynomial null against a smooth alternative.

Usage

rlrt.mp(Y, x = NULL, loginvsp, nbasis = 15, norder = 4, nulldim = NULL,
         evalarg = NULL, get.df = FALSE, B = NULL, P = NULL)

Arguments

Y  ordinarily, an $n \times V$ outcome matrix, where $V$ is the number of hypotheses (in brain imaging applications, the number of voxels). Can also be given by an object of class "fd".

x  a vector or matrix of covariates.

nbasis  number of B-spline basis functions.

norder  order of B-splines.

nulldim  dimension of the null space of the penalty.

loginvsp  a grid of candidate values of the log inverse smoothing parameter.

evalarg  if $Y$ is of class "fd", the argument values at which the functions are evaluated.

get.df  logical: Should the effective df of the smooth at each point be obtained?

B  evaluation matrix of the B-spline basis functions.

P  penalty matrix.

Details

The RLRsim package of Scheipl et al. (2008) is used to simulate the common null distribution of the RLRT statistics.
Value

A list with components

- `table`: matrix of log restricted likelihood ratio values at each grid point, for each test.
- `stat`: RLRT statistics, i.e., the supremum of the values in `table` for each test.
- `logsp`: log smoothing parameter at which the supremum of the restricted likelihood ratio is attained for each test.
- `df`: if `get.df = TRUE`, the effective degrees of freedom corresponding to the log smoothing parameter values in `logsp`.
- `sim`: values simulated from the null distribution of the restricted likelihood ratio statistic.
- `pvalue`: p-values for the RLRT statistics.
- `fdr`: Benjamini-Hochberg false discovery rates corresponding to the above p-values.
- `call`: the call to the function.

Author(s)

Lei Huang <huangracer@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>

References


See Also

- `rlrt4d`, and `Fdr.rlrt` for a more sophisticated false discovery rate procedure.

Examples

```r
Y = matrix(rnorm(6000), nrow=20)
x = rnorm(20)
t4 = rlrt.mp(Y, x, loginvsp = -22:0)
f4 = Fdr.rlrt(t4, 6)
```
Massively parallel restricted likelihood ratio tests (internal)

Description
Conducts a possibly very large number of restricted likelihood ratio tests (Crainiceanu and Ruppert, 2004), with specified random-effects design matrix and fixed-effects design matrix, for a polynomial null against a smooth alternative.

Usage
rlrt.mp.fit(Y, X, Z, loginvsp, evalarg = NULL, get.df = FALSE)

Arguments
Y ordinarily, an $n \times V$ outcome matrix, where $V$ is the number of hypotheses (in brain imaging applications, the number of voxels
X the fixed-effects design matrix.
Z the random-effects design matrix.
loginvsp a grid of candidate values of the log inverse smoothing parameter.
evalarg if Y is of class "fd", the argument values at which the functions are evaluated.
get.df logical: Should the effective df of the smooth at each point be obtained?

Details
The RLRsim package of Scheipl et al. (2008) is used to simulate the common null distribution of the RLRT statistics.

Value
A list with components
table matrix of log restricted likelihood ratio values at each grid point, for each test.
stat RLRT statistics, i.e., the supremum of the values in table for each test.
logsp log smoothing parameter at which the supremum of the restricted likelihood ratio is attained for each test.
df if get.df = TRUE, the effective degrees of freedom corresponding to the log smoothing parameter values in logsp.
sim values simulated from the null distribution of the restricted likelihood ratio statistic.
pvalue p-values for the RLRT statistics.
fdr Benjamini-Hochberg false discovery rates corresponding to the above p-values.
call the call to the function.
Author(s)
Lei Huang <huangracer@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>

References

Examples

```r
Y = matrix(rnorm(6000), nrow=20)
x = rnorm(20)
z = rep(1:5, each = 4)
t4. = rlrt.mp.fit(Y, x, z, loginbsp = -22:0)
```

rlrt4d

Voxelwise restricted likelihood ratio tests

Description
A wrapper function for `rlrt.mp` to handle 3D image responses.

Usage

```r
rlrt4d(arr4d, x = NULL, nbasis = 15, norder = 4, nulldim = NULL, loginbsp, get.df = FALSE, B = NULL, P = NULL)
```

Arguments

- `arr4d` a 4-dimensional response array, where the first 3 dimensions refer to spatial coordinates and the last dimension corresponds to different images.
- `x`, `nbasis`, `norder`, `nulldim`, `loginbsp`, `get.df`, `B`, `P`
  see `rlrt.mp`.

Value

A massively parallel RLRT object, as produced by `rlrt.mp`.

Author(s)
Lei Huang <huangracer@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>

See Also

`plot.rlrt4d`, `rlrt.mp`
Examples

data(test)
d4 = test$d4
x = test$x
rlrtobj = rlrt4d(d4, x, loginvsp = -5:5)
plot(rlrtobj, d4, slice=5)
rlrtpanel(rlrtobj, d4, x)

rlrtpanel

Interactive visualization of voxelwise RLRT results

Description

This function uses interactive graphics tools, provided by the rpanel package, to visualize the results of massively parallel RLRT results as produced by rlrt4d.

Usage

rlrtpanel(rlrtobj, array4d, predictor, disp = c("stat", "p", "fdr"),
titl = "", xlab = "Age", ylab = "", slice = (dim(array4d)[3] %/% 2),
ylim.scatter = NULL, col.image = femmecol(100),
neglog10 = FALSE, threshold = NULL)

Arguments

rlrtobj an RLRT object as produced by rlrt4d.
array4d a 4-dimensional array containing the raw data that were smoothed at each point. The first 3 dimensions refer to x, y, and z coordinates and the last dimension corresponds to different images.
predictor a vector or matrix of covariates.
disp values from the RLRT object to be displayed: either RLRT statistics, p-values, or FDR values.
titl title of the panel.
xlab, ylab x- and y-axis labels.
slice index of the default slice to be shown in the panel.
ylim.scatter the y limits of the scatterplots.
col.image a list of colors such as that generated by rainbow, heat.colors, topo.colors, terrain.colors or similar functions.
neglog10 logical; if TRUE, negative base 10 logarithm (of the quantity specified by disp) is displayed.
threshold the upper limit of the values to be plotted. All larger values will be replaced by the threshold value.
Details

RLRT results are shown for a cross-section (slice), and a slider allows viewing of different slices. Clicking on a voxel produces a scatterplot of the data at that voxel.

Author(s)

Lei Huang <huangracer@gmail.com>

References


See Also

rlrt4d

Examples

# Please see example for rlrt4d

---

screen.vox Screen voxels for a voxelwise smoothing object

Description

Inputs a voxelwise smoothing object as produced by semipar4d, and outputs an object containing the results for a subset of the voxels.

Usage

screen.vox(semi.obj, arr4d, include)

Arguments

semi.obj an object of class semipar.mp.
arr4d the 4-dimensional array used to generate the object.
include a logical matrix indicating which points (or voxels) should be included.

Value

a modified version of semipar.obj, with pointwise coefficients (coef component), pointwise degrees of freedom (pwdf), pointwise log smoothing parameter (pwlsp), and pointwise variance estimate (sigma2) for the points specified by include only.

Author(s)

Lei Huang <huangracer@gmail.com> and Philip Reiss <phil.reiss@nymc.org>
See Also

semipar.mp

Examples

data(test)
d4 = test$d4
x = test$x
vw.obj = semipar4d(d4, formula = ~sf(x), data = data.frame(x = x), lsp=-5:5)

# Include only the first 600 voxels
sv = screen.vox(vw.obj, d4, rep(1:0, c(6:0,4:0)))

Description

Fits a set of semiparametric mixed models, with a common design matrix, by repeated calls to
\texttt{gamm4}. Only a single smooth term is permitted.

Usage

\texttt{semipar.mix.mp(Y, x, param = NULL, random, data.ran, k = 10, norder = 4,
pen.order = 2, knots = "quantile", store.gamm4 = FALSE)}

Arguments

\quad \texttt{Y} \quad n \times V \text{ response matrix.}
\quad \texttt{x} \quad \text{a vector giving the predictor upon which each column of \texttt{Y} is regressed.}
\quad \texttt{param} \quad \text{a matrix or vector for the parametric terms in the model.}
\quad \texttt{random} \quad \text{a formula, passed to \texttt{gamm4}, specifying the random effects structure in \texttt{lmer}
style. See the example.}
\quad \texttt{data.ran} \quad \text{a required data frame containing the factors used for random effects.}
\quad \texttt{k} \quad \text{number of knots.}
\quad \texttt{norder} \quad \text{order of B-splines: the default, 4, gives cubic B-splines.}
\quad \texttt{pen.order} \quad \text{order of the derivative penalty.}
\quad \texttt{knots} \quad \text{knot placement for the B-spline bases. The default, "quantile", gives knots at equally spaced quantiles of the data. The alternative, "equispaced", gives equally spaced knots.}
\quad \texttt{store.gamm4} \quad \text{logical: should the \texttt{gamm4} objects to be stored in the output? FALSE by default.}
Details

Unlike `semipar.mp`, this function does not use large matrix multiplications to avoid looping through model fits. Instead it performs a separate call to `gamm4` to fit a semiparametric mixed model for each column of \( Y \).

Value

- `coef`: matrix of the coefficients obtained from `gamm4` looping (including both parametric and nonparametric parts).
- `bsplinecoef`: matrix of B-spline coefficients.
- `pwdf`: vector of pointwise effective degrees of freedom.
- `pwlsp`: vector of pointwise log smoothing parameters: grid values maximizing the restricted likelihood at each point.
- `B`: matrix of basis function values.
- `C`: the constraint matrix.
- `Z`: transformation matrix to impose constraints.
- `basis`: B-spline basis object, of the type created by the `fda` package; the coefficient estimates are with respect to this basis.

Author(s)

Yin-Hsiu Chen <enjoychen0701@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>

Examples

```r
Y = matrix(rnorm(3),,3)
x1 = rnorm(1)
x2 = matrix(rnorm(2),,2)
family.fac <- factor(rep(1:2,rep(5,2)))
person.fac <- factor(rep(rep(1:25,rep(2,25)),rep(2,5)))
semimix = semipar.mix.mp(Y = Y, x = x1, param = x2, random = ~(1|a/b), data.ran = data.frame(a = family.fac, b = person.fac))
```

Description

Fits a possibly very large number of semiparametric models by quadratically penalized least squares. The model may include a combination of parametric terms, smooth terms, varying-coefficient terms, and simple random effect structures.

Usage

```r
semipar.mp(formula, Y, lsp, data = NULL, range.basis = NULL, knots = "quantile", rm.constr = FALSE, random = NULL, store.reml = FALSE, store.fitted = FALSE)
```
Arguments

formula a formula object such as "~ x1 + sf(x2) + nf(x2, effect = x3)" where x1 is a linear (parametric) predictor, x2 is a predictor on which the responses depend smoothly, and x3 is a predictor whose effect is linear but varies smoothly with x2 (i.e., a varying-coefficient predictor).

Y an n × V response matrix, where V is the number of models fitted in parallel, e.g., voxels in neuroimaging applications.

lsp vector of candidate log tuning parameters (log(λ)).

data an optional data frame containing the variables in the model.

range.basis a numeric vector of length 2 defining the interval over which the B-spline basis is created. If NULL, it will be set as the range of the variable to be evaluated by the basis.

knots knot placement for the B-spline bases. The default, "quantile", gives knots at equally spaced quantiles of the data. The alternative, "equispaced", gives equally spaced knots.

rm.constr logical: should the constraints be removed for varying-coefficient models?

random a formula or a matrix for random effects.

store.reml logical: should the pointwise REML criterion at each grid point be included in the output? FALSE by default, as this output can be very large.

store.fitted logical: should the fitted values be included in the output? FALSE by default.

Details

The basic approach to massively parallel smoothing is described in Reiss et al. (2013). Although simple mixed-effect models are available, semipar.mix.mp is generally preferable for mixed models with a single smooth term.

Each element of list.all corresponding to a nonparametric term of the model is a list with components modmat, penmat, pen.order, start, and end. For each parametric term, the same five components are included, plus basis, argvals, effect, k, and norder.

Value

An object of class "semipar.mp", which is also of class "qplsc.mp" but includes the following additional elements:

where.sf, where.nsf vectors or scalars identifying where the smooth and non-smooth terms, respectively, appear in the model formula.

list.all a list of lists, one for each term of the model; see Details.

formula model formula.

Y response matrix.

lsp candidate values for the log smoothing parameter.

data the supplied data frame, if any.
**Author(s)**

Yin-Hsiu Chen <enjoychen0701@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>

**References**


**Examples**

```r
n<-32
t Ys <- matrix(0, n, 5)
for(i in 1:n) Ys[i,]<--2:2+rnorm(5, i^2, i^0.5)+sin(i)
x1 <- rnorm(n,0,5)
x2 <- 1:n+runif(n, 1, 2)
semipar.obj <- semipar.mp(~x1+sf(x2,k=1),Y=Ys,lsp=seq(5,50,,30))
```

---

**semipar4d**  
*Massively parallel semiparametric regression for 4-dimensional data*

**Description**

This is a wrapper function for `semipar.mp` to handle 3D image responses.

**Usage**

```r
semipar4d(arr4d, formula, lsp, data, range.basis = NULL, knots = "quantile", rm.constr = FALSE, random = NULL, store.reml = FALSE, store.fitted = FALSE)
```

**Arguments**

- `arr4d`  
a 4-dimensional response array, where the first 3 dimensions refer to spatial coordinates and the last dimension corresponds to different images.

- `formula`, `lsp`, `data`, `range.basis`, `knots`, `rm.constr`, `random`, `store.reml`, `store.fitted`

  See `semipar.mp`.

**Value**

An object of class "`semipar.mp`", with two changes. (1) If `store.fitted = TRUE`, the fitted values are given as a 4-dimensional array. (2) A `call` component is included.

**Author(s)**

Yin-Hsiu Chen <enjoychen0701@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>
**sf**

*Defining smooth functions in semiparametric model formulae*

**Description**

This function is called by `semipar.mp` to define B-spline smooths.

**Usage**

```r
sf(argvals, effect = NULL, k = 10, norder = 4, pen.order = 2, range.basis = NULL,
   knots = "quantile")
```

**Arguments**

- `argvals` a vector or matrix of covariates.
- `effect` predictor whose effect varies with respect to `argvals`. E.g., if the effect of diagnosis varies with age, use `sf(age, effect = diagnosis)`. Similar to `argument by in s`.
- `k` number of B-spline basis functions.
- `norder` order of B-splines: the default, 4, gives cubic B-splines.
- `pen.order` order of the penalty, i.e., of the derivative defining the penalty.
- `range.basis` a numeric vector of length 2 defining the interval over which the B-spline basis is created. If NULL, set to the range of the variable.
- `knots` knots placement method for B-spline smoothing. The default, "quantile", places the knots at equally spaced quantiles of the data; "equispaced" gives equally spaced knots.

**Author(s)**

Yin-Hsiu Chen <enjoychen0701@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>
Summary.lm.mp

Summarizing massively parallel linear model fits

Description

summary method for class "lm.mp".

Usage

## S3 method for class 'lm.mp'
summary(object, ...)

Arguments

object an object of class lm.mp, ordinarily created by the function of that name or by lm4d.

... not currently used.

Value

tstat matrix of pointwise t-statistics for each coefficient in the linear model
pvalue matrix of the pointwise p-values for each coefficient in the linear model
aicc vector of pointwise corrected AIC

Author(s)

Philip Reiss <phil.reiss@nyumc.org> and Lei Huang <huangracer@gmail.com>

See Also

lm.mp

Examples

Y = matrix(rnorm(6000), nrow=20)
X = rnorm(20)
tl = lm.mp(Y, ~X)
stl = summary(tl)
**Description**

A randomly generated data set consisting of 50 "response" images and 50 scalar "predictors".

**Usage**

data(test)

**Format**

A list with two components:

- `d4` a $10 \times 10 \times 10 \times 50$ array of responses
- `x` a vector of 50 predictor values

---

**vows-internal**

*Internal functions for the vows package*

**Description**

These functions are ordinarily not to be called by the user, but if you contact the authors with any questions about them, we’ll do our best to clarify matters.

**Author(s)**

Lei Huang <huangracer@gmail.com>, Yin-Hsiu Chen <enjoychen0701@gmail.com>, and Philip Reiss <phil.reiss@nyumc.org>

---

**vows-mgcv**

*Utility functions related to the mgcv package*

**Description**

These internal functions are used by `semipar.mix.mp` (but can also be used more generally) to customize the implementation of B-spline smoothing by `gam`. Specifically, a B-spline smooth with equi-spaced knots can be incorporated in a call to `gam` using a term of the form `s(x, bs="be")`, whereas knots at equally spaced quantiles of the data can be specified by `s(x, bs="bq")`. 
Usage

smooth.construct.be.smooth.spec(object, data, knots)
smooth.construct.bq.smooth.spec(object, data, knots)
Predict.matrix.bspline.smooth(object, data)

Arguments

object a gam smooth specification object generated by a term such as s(x,bs="be") or s(x,bs="bq").
data For smooth.construct.be.smooth.spec and smooth.construct.bq.smooth.spec, a list containing just the data (including any by variable) required by the given term, with names corresponding to object$term (and object$by). The by variable is the last element. For Predict.matrix.bspline.smooth, a data frame containing the values of the (named) covariates at which the smooth term is to be evaluated. Exact requirements are as for smooth.construct and smooth.construct2.
knots a list containing any knots supplied for basis setup, in the same order and with the same names as data. If NULL, a default set of knots is used.

Details

These functions are not normally called directly. For further details, please see smooth.construct.ps.smooth.spec and Predict.matrix.cr.smooth.

Value

Either smooth.construct.be.smooth.spec or smooth.construct.bq.smooth.spec produces an object of class "bspline.smooth": see smooth.construct for the elements that this object will contain. Predict.matrix.bspline.smooth produces a matrix mapping the coefficients for the smooth term to its values at the supplied data values.

Author(s)

Yin-Hsiu Chen <enjoychen0701@gmail.com> and Philip Reiss <phil.reiss@nyumc.org>

Examples

x. = rnorm(20)
smoo.be <- smooth.construct.be.smooth.spec(s(x), data.frame(x = x.), NULL)
smoo.bq <- smooth.construct.bq.smooth.spec(s(x), data.frame(x = x.), NULL)
Predict.matrix.bspline.smooth(smoo.bq, data.frame(x = seq(min(x.),max(x.),100)))
Index

*Topic **datasets**
  test, 29

*Topic **package**
  vows-package, 2

F.mp, 3, 12
fd, 5, 6, 12, 17
Fdr.rlrt, 4, 18
funkmeans, 5, 6–8, 12, 13
funkmeans4d, 6, 6, 7, 8
funkpanel, 2, 6, 7, 7, 13

gam, 29, 30
gamm4, 23, 24
get.ind (vows-internal), 29

heat.colors, 13

kmeans, 5

lm.mp, 2, 3, 8, 9, 10, 28
lm4d, 2, 9, 9, 28
lmer, 23

nii2R, 10, 17

pca.fd, 5
permF.mp, 3, 11
plot, 12–14
plot.funkmeans, 8, 12
plot.rlrt4d, 13, 20
plot.sempar.mp, 14
Predict.matrix.bspline.smooth
  (vows-mgcv), 29
Predict.matrix.cr.smooth, 30

qplsc.mp, 15, 25

R2nii, 11, 16
rainbow, 13
rlr.xz (vows-internal), 29

rlrt.fit (vows-internal), 29
rlrt.mp, 2, 4, 17, 20
rlrt.mp.fit, 19
rlrt4d, 2, 4, 13, 14, 18, 20, 21, 22
rlrtpanel, 2, 13, 14, 21

s, 27
screen.vox, 22

sempar.mlc.mp, 23, 25, 29
sempar.mp, 2, 7, 14, 15, 22–24, 24, 26, 27
sempar4d, 2, 7, 22, 26
sf, 27
smooth.construct, 30
smooth.construct.be.smooth.spec
  (vows-mgcv), 29
smooth.construct.bq.smooth.spec
  (vows-mgcv), 29
smooth.construct.ps.smooth.spec, 30
smooth.construct2, 30
summary.lm.mp, 9, 28

test, 29

vec2img (vows-internal), 29
vows (vows-package), 2
vows-internal, 29
vows-mgcv, 29
vows-package, 2