Package 'hdi'

August 11, 2014

Type Package

Title High-Dimensional Inference

Version 0.1-2

Date 2014-06-30

Author Lukas Meier, Nicolai Meinshausen, Ruben Dezeure

Maintainer Lukas Meier <meier@stat.math.ethz.ch>

Description Implementation of multiple approaches to perform inference in high-dimensional models

Depends glmnet,linprog,parallel,MASS,scalreg

License GPL

Repository CRAN

Repository/R-Forge/Project hdi

Repository/R-Forge/Revision 84

Repository/R-Forge/DateTimeStamp 2014-08-08 13:18:33

Date/Publication 2014-08-11 10:29:47

NeedsCompilation no

R topics documented:

`hdi-package` *hdi*

Description

Perform inference in high-dimensional (generalized) linear models using various approaches.

Details

Package:	hdi
Type:	Package
Version:	1.0
Date:	2013-04-02
License:	GPL

This is a very early test release!

Author(s)

Lukas Meier

Maintainer: Lukas Meier <meier@stat.math.ethz.ch>

References

Meinshausen, N., Meier, L. and B\"uhlmann, P. (2009), *P-values for high-dimensional regression*, Journal of the American Statistical Association 104, 1671-1681.

Meinshausen, N. and B\"uhlmann, P. (2010), *Stability selection (with discussion)*, Journal of the Royal Statistical Society: Series B, 72, 417-473.

clusterGroupBound *Group test of variable importance in a high-dimensional linear model, using a hierarchical structure.*

Description

Computes confidence intervals for the l1-norm of groups of regression parameters in a hierarchical clustering tree.

Usage

```
clusterGroupBound(x, y, method = "average",
                  dist = as.dist(1 - abs(cor(x))), alpha = 0.05,
                  hcloutput, nsplit = 11, s = min(10, ncol(x) - 1),
                  silent = FALSE, setseed = TRUE, lpSolve = TRUE)
```

Arguments

x	The design matrix of the regression with p columns for p predictor variables and n rows that correspond to n observations.
y	The response variable; a numeric vector of length n.
method	The method used for constructing the hierarchical clustering tree (default is "average" linkage). Alternatively, you can provide your own hierarchical clustering through the optional argument hcloutput.
dist	A distance matrix can be entered as an argument, on which the hierarchical clustering will be based. The default option is that the distance between variables will be calculated as 1 less the absolute correlation matrix. Alternatively, you can provide your own hierarchical clustering through the optional argument hcloutput.
alpha	The level in (0, 1) at which the confidence intervals are to be constructed.
hcloutput	Optional argument. The output of a call the the hclust function. If it is provided, the arguments dist and method are ignored.
nsplit	The number of data splits used.
s	The dimensionality of the projection that is used. Lower values lead to faster computation and if n > 50, then s is set to 50 if left unspecified to avoid lengthy computations.
silent	Output is supressed if this option is set to true.
setseed	If setseed is true (recommended), then the same random seeds are used for all groups, which makes the confidence intervals simultaneously valid over all groups of variables tested.
lpSolve	Only set to false if lpSolve is not working on the current machine (setting it to false will results in much slower computations; only use on small problems).

Value

Returns a list with components

groupNumber	The index of the group tested in the original hierarchical clustering tree
members	A list containing the variables that belong into each testes group
noMembers	A vector containing the number of members in each group
lowerBound	The lower bound on the l1-norm in each group
position	The position on the x-axis of each group (used for plotting)
leftChild	Gives the index of the group that corresponds to the left child node in the tested tree (negative values correspond to leaf nodes)
rightChild	Same as leftCHild for the right child of each node
isLeaf	Logical vector. Is TRUE for a group if it is a leaf node in the tested tree or if both child nodes have a zero lower bound on their group l1-norm

Author(s)

Nicolai Meinshausen meinshausen@stat.math.ethz.ch

References

Nicolai Meinshausen (2013) Assumption-free confidence intervals for groups of variables in sparse high-dimensional regression. http://arxiv.org/abs/1309.3489

See Also

Use clusterGroupBound to test all groups in a hierarchical clustering tree. Use groupBound to compute the lower bound for selected groups of variables.

Examples

```
## Create a regression problem with block-design: p = 10, n = 30,
## block size B = 5 and within-block correlation of rho = 0.99
p   <- 10
n   <- 100
B   <- 5
rho <- 0.99

ind <- rep(1:ceiling(p / B), each = B)[1:p]
Sigma <- diag(p)

for (ii in unique(ind)){
  id <- which(ind == ii)
  Sigma[id, id] <- rho
}
diag(Sigma) <- 1

x <- matrix(rnorm(n * p), nrow = n) %*% chol(Sigma)

## Create response with active variables 1 and 21
```

```
beta    <- rep(0, p)
beta[1] <- 5

y  <- as.numeric(x %*% beta + rnorm(n))

out <- clusterGroupBound(x, y, nsplit = 5)

## Plot and print the hierarchical group-test
plot(out)
print(out)
```

`fdr.adjust` *Function to calculate FDR adjusted p-values*

Description

Calculates FDR adjusted p-values similar to R-function p.adjust but *without* adjustment for multiplicity.

Usage

```
fdr.adjust(p)
```

Arguments

p	Vector of p-values.

Details

It is assumed that the p-values are already corrected for multiplicity. P-values with a value of 1 are currently ignored.

Value

Vector of p-values.

Author(s)

Lukas Meier

References

Meinshausen, N., Meier, L. and B\"uhlmann, P. (2009), *P-values for high-dimensional regression*, Journal of the American Statistical Association 104, 1671-1681.

See Also

`p.adjust`

Examples

```
x <- matrix(rnorm(100*1000), nrow = 100, ncol = 1000)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)

## Multi-splitting with lasso.firstq as model selector function
fit.multi <- multi.split(x, y, model.selector =lasso.firstq,
                         args.model.selector = list(q = 10))
p.adjust <- fdr.adjust(fit.multi$pval.corr)
```

`glm.pval` *Function to calculate p-values for a generalized linear model.*

Description

Calculates (classical) p-values for an ordinary generalized linear model in the n > p situation.

Usage

```
glm.pval(x, y, family = "binomial", trace = FALSE, ...)
```

Arguments

`x`	Design matrix (without intercept).
`y`	Response vector.
`family`	As in `glm`.
`trace`	Logical. Should information be printed out if algorithm did not converge?
`...`	Additional arguments to be passed to `glm`.

Details

A model with intercept is fitted but the p-value of the intercept is not reported in the output.

Value

Vector of p-values (not including the intercept).

Author(s)

Lukas Meier

See Also

`hdi`

Examples

```
## ...
```

groupBound *Lower bound on the l1-norm of groups of regression variables*

Description

In a (high-dimensional) regression, the function returns a lower bound that forms a one-sided confidence interval for the group l1-norm of a specified group of regression parameters. It is assumed that errors have a Gaussian distribution with unknown noise level. The underlying vector that inference is made about is the l1-sparsest approximation to the noiseless data. Under a weak compatbility condition, this is identical to inference about the l1-sparsest approximation to the noiseless data.

Usage

```
groupBound(x, y, group, alpha = 0.05, nsplit = 11,
                s = min(10, ncol(x) - 1), setseed = TRUE,
                silent = FALSE, lpSolve = TRUE,
                parallel = FALSE, ncores = 4)
```

Arguments

x	The design matrix of the regression with p columns for p predictor variables and n rows that correspond to n observations.
y	The response variable; a numeric vector of length n.
group	Either a numeric vector with entries in 1,...,p or a list with such numeric vectors. If group is just a numeric vector, this is the group of variables for which a lower bound is computed. If group is a list, the lower bound is computed for each group in the list.
alpha	The level at which the test/ confidence interval is computed; a numeric value in (0,1).
nsplit	The number of data splits used.
s	The dimensionaility of the projection that is used. Lower values lead to faster computation and if n>50, then s is set to 50 if left unspecified to avoid lengthy computations.
setseed	If setseed is true (recommended), then the same random seeds are used for all groups, which makes the confidence intervals simulatenously valid over all groups of variables tested.
silent	Output is supressed if this option is set to true.
lpSolve	Only set to false if lpSolve is not working on the current machine (setting it to false will results in much slower computations; only use on small problems).
parallel	Should parallelization be used? (logical)
ncores	Number of cores used for parallelization.

Details

The data are split since the noise level is unknown. On the first part of the random split, a cross-validated lasso solution is computed, using the glmnet implementation. This estimator is used as an initial estimator on the second half of the data. Results at level alpha are aggregated over `nsplit` splits via the median of results at levels alpha/2.

Value

If `group` is a single numeric vector, a scalar containg the lower bound for this group of variables is returned. If `group` is a list, a numeric vector is retuned where each entry corresponds to the group of variables defined in the same order in `group`.

Author(s)

Nicolai Meinshausen meinshausen@stat.math.ethz.ch

References

Nicolai Meinshausen (2013) Assumption-free confidence intervals for groups of variables in sparse high-dimensional regression. http://arxiv.org/abs/1309.3489

See Also

Use `clusterGroupBound` to test all groups in a hierarchical clustering tree.

Examples

```
## Create a regression problem with block-design: p = 10, n = 30,
## block size B = 5 and within-block correlation of rho = 0.99
p   <- 10
n   <- 100
B   <- 5
rho <- 0.99

ind <- rep(1:ceiling(p / B), each = B)[1:p]
Sigma <- diag(p)

for (ii in unique(ind)){
  id <- which(ind == ii)
  Sigma[id, id] <- rho
}
diag(Sigma) <- 1

x <- matrix(rnorm(n * p), nrow = n) %*% chol(Sigma)

## Create response with active variables 1 and 21
beta    <- rep(0, p)
beta[1] <- 5

y  <- as.numeric(x %*% beta + rnorm(n))
```

```
## Compute lower bounds:

## Lower bound for the l1-norm of all variables 1-10 of the sparsest
## optimal vector
lowerBoundAll <- groupBound(x, y, 1:p)
print(lowerBoundAll)
cat("\nlower bound for all variables 1-10: ", lowerBoundAll, "\n")

## Compute lower bounds:
## Lower bounds for variable 1 in itself, then groups 1-5
lowerBound <- groupBound(x, y, list(1, 1:5))
cat("lower bound for the groups {1}, {1,...,5}: ", lowerBound, "\n")
```

`hdi` *Function to perform inference in high-dimensional (generalized) linear models*

Description

Perform inference in high-dimensional (generalized) linear models using various approaches.

Usage

```
hdi(x, y, method = "multi.split", B = NULL, fraction = 0.5,
    model.selector = NULL, EV = NULL, threshold = 0.75,
    gamma = seq(0.05, 0.99, by = 0.01),
    classical.fit = NULL,
    args.model.selector = NULL, args.classical.fit = NULL,
    trace = FALSE, ...)
```

Arguments

`x`	Design matrix (without intercept).
`y`	Response vector.
`method`	Multi-splitting ("multi.split") or stability-selection ("stability").
`B`	Number of sample-splits (for "multi.split") or sub-sample iterations (for "stability"). Default is 50 ("multi.split")or 100 ("stability"). Ignored otherwise.
`fraction`	Fraction of data used at each of the B iterations.
`model.selector`	Function to perform model selection. Default is `lasso.cv` ("multi.split") and `lasso.firstq` ("stability"). Function must have at least two arguments: x (the design matrix) and y (the response vector). Return value is the index vector of selected columns. See `lasso.cv` and `lasso.firstq` for examples. Additional arguments can be passed through `args.model.selector`.
`EV`	(only for "stability"). Bound(s) for expected number of false positives . Can be a vector.
`threshold`	(only for "stability"). Bound on selection frequency.

gamma	(only for "multi.split"). Vector of gamma-values.
classical.fit	(only for "multi.split"). Function to calculate (classical) p-values. Default is lm.pval. Function must have at least two arguments: x (the design matrix) and y (the response vector). Return value is the vector of p-values. See lm.pval for an example. Additional arguments can be passed through args.classical.fit.
args.model.selector	Named list of further arguments for function model.selector.
args.classical.fit	Named list of further arguments for function classical.fit.
trace	Should information be printed out while computing (logical).
...	Other arguments to be passed to the underlying functions.

Value

pval	(only for "multi.split"). Vector of p-values.
gamma.min	(only for "multi.split"). Value of gamma where minimal p-values was attained.
select	(only for "stability"). List with selected predictors for the supplied values of EV.
EV	(only for "stability"). Vector of corresponding values of EV.
thresholds	(only for "stability"). Used thresholds.
freq	(only for "stability"). Vector of selection frequencies.

Author(s)

Lukas Meier

References

Meinshausen, N., Meier, L. and B\"uhlmann, P. (2009), *P-values for high-dimensional regression*, Journal of the American Statistical Association 104, 1671-1681.

Meinshausen, N. and B\"uhlmann, P. (2010), *Stability selection (with discussion)*, Journal of the Royal Statistical Society: Series B, 72, 417-473.

See Also

stability, multi.split

Examples

```
x <- matrix(rnorm(100*1000), nrow = 100, ncol = 1000)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)

## Multi-splitting with lasso.firstq as model selector function
fit.multi <- hdi(x, y, method = "multi.split",
                 model.selector =lasso.firstq,
                 args.model.selector = list(q = 10))
fit.multi
fit.multi$pval.corr[1:10] ## the first 10 p-values
```

```
## Stability selection
fit.stab <- hdi(x, y, method = "stability", EV = 2)
fit.stab
fit.stab$freq[1:10] ## frequency of the first 10 predictors
```

`lasso.cv` *Function to select predictors based on 10-fold cross-validation of the lasso estimator.*

Description

Performs (10-fold) cross-validation and determines the prediction optimal set of parameters

Usage

```
lasso.cv(x, y, ...)
```

Arguments

`x`	Design matrix (without intercept).
`y`	Number of predictors that should be selected.
`...`	Further arguments to be passed to `cv.glmnet`.

Details

Function basically only calls cv.glmnet, see source code.

Value

Vector of selected predictors.

Author(s)

Lukas Meier

See Also

`hdi`.

Examples

```
x <- matrix(rnorm(100*1000), nrow = 100, ncol = 1000)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)
sel <- lasso.cv(x, y)
sel
```

`lasso.firstq` *Function to determine the first q predictors in the lasso path.*

Description

Determines the q predictors that enter the lasso path first.

Usage

```
lasso.firstq(x, y, q, ...)
```

Arguments

`x`	Design matrix (without intercept).
`y`	Response vector.
`q`	Number of predictors that should be selected.
`...`	Additional arguments to be passed to `glmnet`.

Details

Function only calls glmnet in a special way, see source code.

Value

Vector of selected predictors.

Author(s)

Lukas Meier

See Also

`hdi`.

Examples

```
x <- matrix(rnorm(100*1000), nrow = 100, ncol = 1000)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)
sel <- lasso.firstq(x, y, q = 5)
sel
```

lasso.proj *P-values based on lasso projection method*

Description

P-values based on lasso projection method

Usage

```
lasso.proj(x, y, family = "gaussian", standardize = TRUE,
           multiplecorr.method = "holm", N = 10000,
           parallel = FALSE, ncores = 4,
           sigma = NULL, Z = NULL)
```

Arguments

x	Design matrix (without intercept).
y	Response vector.
family	family
standardize	Should design matrix be standardized to unit column standard deviation.
multiplecorr.method	Either "WY" or any of p.adjust.methods.
N	Number of empirical samples (only used if multiplecorr.method == "WY")
parallel	Should parallelization be used? (logical)
ncores	Number of cores used for parallelization.
sigma	Estimate of standard deviation of error term.
Z	user input

Value

pval	Individual p-values for each parameter.
pval.corr	Multiple testing corrected p-values for each parameter.
groupTest	Function to perform groupwise tests. Groups are indicated using an index vector with entries in 1,...,p.
clusterGroupTest	Function to perform groupwise tests based on hierarchical clustering. You can either provide a distance matrix and clustering method or the output of hierarchical clustering from the function hclust as for clusterGroupBound.
sigmahat	$\widehat{\sigma}$ coming from the scaled lasso.

Author(s)

Ruben Dezeure

References

van de Geer, S., B\"uhlmann, P., Ritov, Y. and Dezeure, R. (2013). On asymptotically optimal confidence regions and tests for high-dimensional models. Preprint.

Zhang, C., Zhang, S. (2014), Confidence intervals for low dimensional parameters in high dimensional linear models, Journal of the Royal Statistical Society: Series B (Statistical Methodology).

Examples

```
x <- matrix(rnorm(100*20), nrow = 100, ncol = 20)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)
fit.lasso <- lasso.proj(x, y)
which(fit.lasso$pval.corr < 0.05)
```

`lm.ci` *Function to calculate confidence intervals for ordinary multiple linear regression.*

Description

Calculates (classical) confidence intervals for an ordinary multiple linear regression model in the n > p situation.

Usage

```
lm.ci(x, y, level = 0.95, ...)
```

Arguments

`x`	Design matrix (without intercept).
`y`	Response vector.
`level`	Coverage level.
`...`	Additional arguments to be passed to `lm`.

Details

A model with intercept is fitted but the p-value of the intercept is not reported in the output.

Value

Matrix of confidence interval bounds (not including the intercept).

Author(s)

Lukas Meier

See Also

hdi

Examples

```
x <- matrix(rnorm(100*5), nrow = 100, ncol = 5)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)
ci <- lm.ci(x, y)
ci
```

`lm.pval` *Function to calculate p-values for ordinary multiple linear regression.*

Description

Calculates (classical) p-values for an ordinary multiple linear regression in the n > p situation.

Usage

```
lm.pval(x, y, exact = TRUE, ...)
```

Arguments

`x`	Design matrix (without intercept).
`y`	Response vector.
`exact`	Logical. TRUE if p-values based on t-distribution should be calculated. FALSE if normal distribution should be used as approximation.
`...`	Additional arguments to be passed to `lm`.

Details

A model with intercept is fitted but the p-value of the intercept is not reported in the output.

Value

Vector of p-values (not including the intercept).

Author(s)

Lukas Meier

See Also

hdi

Examples

```
x <- matrix(rnorm(100*5), nrow = 100, ncol = 5)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)
pval <- lm.pval(x, y)
pval
```

`multi.split` *Function to calculate p-values based on multi-splitting approach*

Description

Function to calculate p-values based on multi-splitting approach

Usage

```
multi.split(x, y, B = 100, fraction = 0.5, ci = TRUE, ci.level = 0.95,
            model.selector = lasso.cv,
            classical.fit = lm.pval, classical.ci = lm.ci,
            parallel = FALSE, ncores = 4,
            gamma = seq(ceiling(0.05*B)/B,1-1/B,by=1/B),
            args.model.selector = NULL, args.classical.fit = NULL,
            args.classical.ci = NULL,
            return.nonaggr = FALSE, return.selmodels = FALSE, trace = FALSE)
```

Arguments

`x`	Design matrix (without intercept).
`y`	Response vector.
`B`	Number of sample-splits.
`fraction`	Fraction of data used at each sample split for the model selection process. The remaining data is used for calculating the p-values.
`ci`	Should a confidence interval be calculated for every parameter? (logical)
`ci.level`	Coverage level of confidence interval.
`model.selector`	Function to perform model selection. Default is `lasso.cv`. Function must have at least two arguments: x (the design matrix) and y (the response vector). Return value is the index vector of selected columns. See `lasso.cv` and `lasso.firstq` for an example. Additional arguments can be passed through `args.model.selector`.
`classical.fit`	Function to calculate (classical) p-values. Default is `lm.pval`. Function must have at least two arguments: x (the design matrix) and y (the response vector). Return value is the vector of p-values. See `lm.pval` for an example. Additional arguments can be passed through `args.classical.fit`.

`classical.ci` Function to calculate (classical) confidence intervals. Default is `lm.ci`. Function must have at least 3 arguments: x (the design matrix), y (the response vector) and level (the coverage level). Return value is the matrix of confidence intervals. See `lm.ci` for an example. Additional arguments can be passed through `args.classical.ci`.

`parallel` Should parallelization be used? (logical)

`ncores` Number of cores used for parallelization.

`gamma` Vector of gamma-values. In case gamma is a scalar, the value Q_j instead of P_j is being calculated (see reference below).

`args.model.selector` Named list of further arguments for function `model.selector`.

`args.classical.fit` Named list of further arguments for function `classical.fit`.

`args.classical.ci` Named list of further arguments for function `classical.ci`.

`return.nonaggr` Should the unadjusted p-values be reported? (logical).

`return.selmodels` Should the selected models (at each split) be reported? (logical).

`trace` Should information be printed out while computing? (logical).

Details

...

Value

`pval.corr` Vector of multiple testing corrected p-values.

`gamma.min` Value of gamma where minimal p-values was attained.

Author(s)

Lukas Meier

References

Meinshausen, N., Meier, L. and B\"uhlmann, P. (2009), *P-values for high-dimensional regression*, Journal of the American Statistical Association 104, 1671-1681.

See Also

...

Examples

```
x <- matrix(rnorm(100*1000), nrow = 100, ncol = 1000)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)

## Multi-splitting with lasso.firstq as model selector function
fit.multi <- multi.split(x, y, model.selector =lasso.firstq,
                         args.model.selector = list(q = 10))
fit.multi
fit.multi$pval.corr[1:10] ## the first 10 p-values
```

`plot.clusterGroupBound` *Plot output of hierarchical testing of groups of variables*

Description

The functions plots the outcome of applying a lower bound on the l1-norm on groups of variables in a hierarchical clustering tree.

Usage

```
## S3 method for class 'clusterGroupBound'
plot(x, cexfactor = 1, yaxis = "members",
     col = NULL, ...)
```

Arguments

`x`	The output of function `clusterGroupBound`
`cexfactor`	Multiplies the size of the node symbols.
`yaxis`	For the default value ("members"), the hierarchical tree is shown as function of cluster size on the y-axis, whereas the node sizes are proportional to the lower l1-norm of the respective groups of variables. If `yaxis` takes any different value, then this is reversed and the tree is shown against the lower l1-norm on the y-axis, while node sizes are now proportional to the number of elements in each cluster.
`col`	The colour of the symbols for the nodes.
`...`	Additional arguments.

Value

Nothing is returned

Author(s)

Nicolai Meinshausen meinshausen@stat.math.ethz.ch

References

Nicolai Meinshausen (2013) Assumption-free confidence intervals for groups of variables in sparse high-dimensional regression. http://arxiv.org/abs/1309.3489

See Also

Use `clusterGroupBound` to test all groups in a hierarchical clustering tree. Use `groupBound` to compute the lower bound for selected groups of variables.

Examples

```
## Create a regression problem with block-design: p = 10, n = 30,
## block size B = 5 and within-block correlation of rho = 0.99
p   <- 10
n   <- 100
B   <- 5
rho <- 0.99

ind <- rep(1:ceiling(p / B), each = B)[1:p]
Sigma <- diag(p)

for (ii in unique(ind)){
  id <- which(ind == ii)
  Sigma[id, id] <- rho
}
diag(Sigma) <- 1

x <- matrix(rnorm(n * p), nrow = n) %*% chol(Sigma)

## Create response with active variables 1 and 21
beta    <- rep(0, p)
beta[1] <- 5

y  <- as.numeric(x %*% beta + rnorm(n))

## Compute the lower bound for all groups in a hierarchical clustering tree
out <- clusterGroupBound(x, y, nsplit = 5)

## Plot the tree with y-axis proportional to the (log) of the number of
## group members and node sizes proportional to the lower l1-norm bound.
plot(out)

## Show the lower bound on the y-axis and node sizes proportional to
## number of group members
plot(out, yaxis = "")
```

`riboflavin` *Riboflavin data set*

Description

Dataset of riboflavin production by Bacillus subtilis containing $n = 71$ observations of $p = 4088$ predictors (gene expressions) and a one-dimensional response (riboflavin production).

Usage

```
data(riboflavin)
```

Format

y Log-transformed riboflavin production rate (original name: q_RIBFLV).

x (Co-)variables measuring the logarithm of the expression level of 4088 genes.

Details

Data kindly provided by DSM (Switzerland).

References

B\"uhlmann, P., Kalisch, M. and Meier, L. (2013). *High-dimensional statistics with a view towards applications in biology*. To appear in Annual Review of Statistics and its Applications.

Examples

```
data(riboflavin)
```

`ridge.proj` *P-values based on ridge projection method*

Description

P-values based on ridge projection method

Usage

```
ridge.proj(x, y, family = "gaussian", standardize = TRUE,
           lambda = 1, sigma = NULL, multiplecorr.method = "holm",
           N = 10000)
```

Arguments

`x`	Design matrix (without intercept).
`y`	Response vector.
`family`	family
`standardize`	Should design matrix be standardized to unit column standard deviation (logical)?

lambda	Value of penalty parameter lambda (ridge regression).
sigma	Estimate of error standard deviation
multiplecorr.method	Either "WY" or any of p.adjust.methods.
N	Number of empirical samples (only used if multiplecorr.method == "WY")

Value

pval	Individual p-values for each parameter.
pval.corr	Multiple testing corrected p-values for each parameter.
groupTest	Function to perform groupwise tests. Groups are indicated using an index vector with entries in 1,...,p.
clusterGroupTest	Function to perform groupwise tests based on hierarchical clustering. You can either provide a distance matrix and clustering method or the output of hierarchical clustering from the function hclust as for clusterGroupBound.
sigmahat	$\widehat{\sigma}$ coming from the scaled lasso.

Author(s)

Peter Buehlmann, Ruben Dezeure, Lukas Meier

References

B\"uhlmann, P. (2013), *Statistical significance in high-dimensional linear models*, Bernoulli 19, 1212-1242.

Examples

```
x <- matrix(rnorm(100*100), nrow = 100, ncol = 100)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)
fit.ridge <- ridge.proj(x, y)
which(fit.ridge$pval.corr < 0.05)
```

---

stability *Function to perform stability selection*

Description

Function to perform stability selection

Usage

```
stability(x, y, EV, threshold = 0.75, B = 100, fraction = 0.5,
          model.selector = lasso.firstq, args.model.selector = NULL,
          parallel = FALSE, ncores = 4, trace = FALSE)
```

Arguments

x	Design matrix (without intercept).
y	Response vector.
EV	Bound for expected number of false positives.
threshold	Threshold for selection frequency. Must be in (0.5, 1).
B	Number of sub-sample iterations.
fraction	Fraction of data used at each of the B sub-samples.
model.selector	Function to perform model selection. Default is lasso.firstq. User supplied function must have at least three arguments: x (the design matrix), y (the response vector) and q (the maximal model size). Return value is the index vector of selected columns. See lasso.firstq for an example. Additional arguments can be passed through args.model.selector.
args.model.selector	Named list of further arguments for function model.selector.
parallel	Should parallelization be used? (logical)
ncores	Number of cores used for parallelization.
trace	Should information be printed out while computing (logical).

Details

...

Value

selected	Vector of selected predictors.
freq	Vector of selection frequencies.
q	Size of fitted models in order to control error rate at desired level.

Author(s)

Lukas Meier

References

B\"hlmann, P., Kalisch, M. and Meier, L. (2013). *High-dimensional statistics with a view towards applications in biology*. To appear in Annual Review of Statistics and its Applications. Preprint

See Also

...

Examples

```
x <- matrix(rnorm(100*1000), nrow = 100, ncol = 1000)
y <- x[,1] * 2 + x[,2] * 2.5 + rnorm(100)
fit.stab <- stability(x, y, EV = 1)
fit.stab
fit.stab$freq[1:10] ## selection frequency of the first 10 predictors
```

Index