Package 'prototest'

February 3, 2019

Type Package

Title Inference on Prototypes from Clusters of Features

Version 1.2

Date 2019-02-02

Author Stephen Reid

Maintainer Stephen Reid <sreid1652@gmail.com>

Depends intervals, MASS, glmnet

Description Procedures for testing for group-wide signal in clusters of variables. Tests can be performed for single groups in isolation (univariate) or multiple groups together (multivariate). Specific tests include the exact and approximate (un)selective likelihood ratio tests described in Reid et al (2015), the selective F test and marginal screening prototype test of Reid and Tibshirani (2015). User may pre-specify columns to be included in prototype formation, or allow the function to select them itself. A mixture of these two is also possible. Any variable selection is accounted for using the selective inference framework. Options for non-sampling and hit-and-run null reference distributions.

License GPL (>= 2)

Imports Rcpp (>= 0.12.1)

LinkingTo Rcpp, RcppArmadillo

URL http://arxiv.org/abs/1511.07839

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-02-03 11:00:03 UTC

R topics documented:

`prototest-package` *Inference on Prototypes from Clusters of Features*

Description

Procedures for testing for group-wide signal in clusters of variables. Tests can be perfromed for single groups in isolation (univariate) or multiple groups together (multivariate). Specific tests include the exact and approximate (un)selective likelihood ratio (ELR, ALR) tests described in Reid et al (2015), the selective F test and marginal screening prototype test of Reid and Tibshirani (2015). User may prespecify columns to be included in prototype formation, or allow the function to select them itself. A mixture of these two is also possible. Any variable selection is accounted for using the selective inference framework introduced in Lee et al (2013) and further developed in Lee and Taylor (2014). Options for non-sampling and hit-and-run null reference distrbutions. Tests are examples of selected model tests, a notion introduced in Fithian et al (2015).

Details

Package:	prototest
Type:	Package
Version:	1.0
Date:	2015-11-12
License:	GPL (>= 2)

Only two functions provided: `prototest.univariate` (for tests with a single group in isolation) and `prototest.multivariate` (for tests with multiple groups simultaneously). Each function provides options to perform one of the ELR, ALR, F or marginal screening prototype tests. User may specify which columns are to be used in prototype construction, or leave it for the function to select. Valid tests are performed in the event of variable selection. User has option to use non-sampling null reference distributions (where available) or hit-and-run references.

Author(s)

Stephen Reid

Maintainer: Stephen Reid <sreid@stanford.edu>

References

Reid, S. and Tibshirani, R. (2015) *Sparse regression and marginal testing using cluster prototypes*. `http://arxiv.org/pdf/1503.00334v2.pdf`. *Biostatistics doi: 10.1093/biostatistics/kxv049*
Reid, S., Taylor, J. and Tibshirani, R. (2015) *A general framework for estimation and inference from clusters of features*. Available online: `http://arxiv.org/abs/1511.07839`
Lee, J.D., Sun, D.L., Sun, Y. and Taylor, J.E. (2013) *Exact post-selection inference, with application to the lasso*. `http://arxiv.org/pdf/1311.6238v6.pdf`. *Annals of Statistics (to appear)*
Lee, J.D. and Taylor, J.E. (2014) *Exact Post Model Selection Inference for Marginal Screening*. `http://arxiv.org/pdf/1402.5596v2.pdf`

Fithian, W., Sun, D.L. and Taylor, J.E. (2015) *Optimal Inference After Model Selection*. http://arxiv.org/pdf/1410.2597v2.pdf

Examples

```
require (prototest)

### generate data
set.seed (12345)
n = 100
p = 80

X = matrix (rnorm(n*p, 0, 1), ncol=p)


beta = rep(0, p)
beta[1:3] = 0.1 # three signal variables: number 1, 2, 3
signal = apply(X, 1, function(col){sum(beta*col)})
intercept = 3

y = intercept + signal + rnorm (n, 0, 1)

### treat all columns as if in same group and test for signal

# non-selective ELR test with nuisance intercept
elr = prototest.univariate (X, y, ”ELR”, selected.col=1:5)
# selective F test with nuisance intercept; non-sampling
f.test = prototest.univariate (X, y, ”F”, lambda=0.01, hr.iter=0)
print (elr)
print (f.test)

### assume variables occur in 4 equally sized groups
num.groups = 4
groups = rep (1:num.groups, each=p/num.groups)

# selective ALR test -- select columns 21-25 in 2nd group; test for signal in 1st; hit-and-run
alr = prototest.multivariate(X, y, groups, 1, ”ALR”, 21:25, lambda=0.005, hr.iter=20000)
# non-selective MS test -- specify first column in each group; test for signal in 1st
ms = prototest.multivariate(X, y, groups, 1, ”MS”, c(1,21,41,61))
print (alr)
print (ms)
```

print.prototest *Print* prototest *object*

Description

Generic print method for prototest objects

Usage

```
## S3 method for class 'prototest'
 print(x, ...)
```

Arguments

`x`	object of type `prototest`.
`...`	other parameters passed to `print` function.

Details

Prints the test statistic and p-value associated with the `prototest` object `x`.

Author(s)

Stephen Reid

See Also

`prototest.univariate`, `prototest.multivariate`

Examples

```
require (prototest)

### generate data
set.seed (12345)
n = 100
p = 80

X = matrix (rnorm(n*p, 0, 1), ncol=p)


beta = rep(0, p)
beta[1:3] = 2 # three signal variables: number 1, 2, 3
signal = apply(X, 1, function(col){sum(beta*col)})
intercept = 3

y = intercept + signal + rnorm (n, 0, 1)

### treat all columns as if in same group and test for signal

# non-selective ELR test with nuisance intercept
elr = prototest.univariate (X, y, "ELR", selected.col=1:5)
print (elr)
```

prototest.multivariate *Perform Prototype or F tests for Significance of Groups of Predictors in the Multivariate Model*

Description

Perform prototype or F tests for significance of groups of predictors in the multivariate model. Choose either exact or approximate likelihood ratio prototype tests (ELR) or (ALR) or F test or marginal screening prototype test. Options for selective or non-selective tests. Further options for non-sampling or hit-and-run reference distributions for selective tests.

Usage

```
prototest.multivariate(x, y, groups, test.group, type = c("ELR", "ALR", "F", "MS"),
selected.col = NULL, lambda, mu = NULL, sigma = 1,
hr.iter = 50000, hr.burn.in = 5000, verbose = FALSE, tol = 10^-8)
```

Arguments

`x` input matrix of dimension *n*-by-*p*, where *p* is the number of predictors over all predictor groups of interest. Will be mean centered and standardised before tests are performed.

`y` response variable. Vector of length *n*, assumed to be quantitative.

`groups` group membership of the columns of `x`. Vector of length *p*, which each element containing the goup label of the corresponding column in `x`.

`test.group` group label for which we test nullity. Should be one of the values seen in `groups`. See Details for further explanation.

`type` type of test to be performed. Can select one at a time. Options include the exact and approximate likelihood ratio prototype tests of Reid et al (2015) (ELR, ALR), the F test and the marginal screening prototype test of Reid and Tibshirani (2015) (MS). Default is ELR.

`selected.col` preselected columns selected by the user. Vector of indices in the set {1, 2, ... *p*}. Used in conjunction with `groups` to ascertain for which groups the user has specified selected columns. Should it find any selected columns within a group, no further action is taken to select columns. Should no columns within a group be specified, columns are selected using either lasso or the marginal screening procedure, depending on the test. If all groups have prespecified columns, a non-selective test is performed, using the classical distributional assumptions (exact and/or asymptotic) for the test in question. If any selection is performed, selective tests are performed. Default is `NULL`, requiring the selection of columns in all the groups.

`lambda` regularisation parameter for the lasso fit. Same for each group. Must be supplied when at least one group has unspecified columns in `selected.col`. Will be supplied to `glmnet`. This is the unstandardised version, equivalent to `lambda/n` supplied to `glmnet`.

`mu` mean parameter for the response. See Details below. If supplied, it is first subtracted from the response to yield a zero-mean (at the population level) vector for which we proceed with testing. If `NULL` (the default), this parameter is treated as nuisance parameter and accounted for as such in testing.

`sigma` error standard deviation for the response. See Details below. Must be supplied. If not, it is assumed to be 1. Required for computation of some of the test statistics.

`hr.iter` number of hit-and-run samples required in the reference distribution of the a selective test. Applies only if `selected.col` is `NULL`. Default is 50000. Since dependent samples are generated, large values are required to generate good reference distributions. If set to 0, the function tries to applu a non-sampling selective test (provided `selected.col` is `NULL`), if possible. If non-sampling test is not possible, the function exits with a message.

`hr.burn.in` number of burn-in hit-and-run samples. These are generated first so as to make subsequent hit-and-run realisations less dependent on the observed response. Samples are then discarded and do not inform the null reference distribution.

`verbose` should progress be printed?

`tol` convergence threshold for iterative optimisation procedures.

Details

The model underpinning each of the tests is

$$y = \mu + \sum_{k=1}^{K} \theta_k \cdot \hat{y}_k + \epsilon$$

where $\epsilon \sim N(0, \sigma^2 I)$ and *K* is the number of predictor groups. $\hat{y}_k$ depends on the particular test considered.

In particular, for the ELR, ALR and F tests, we have $\hat{y}_k = P_{M_k}(y - \mu)$, where $P_{M_k} = X_{M_k}\left(X_{M_k}^\top X_{M_k}\right)^{-1} X_{M_k}^\top$. X_M is the input matrix reduced to the columns with indices in the set *M*. M_k is the set of indices selected from considering group *k* of predictors in isolation. This set is either provided by the user (via `selected.col`) or is selected automatically (if `selected.col` is `NULL`). If the former, a non-selective test is performed; if the latter, a selective test is performed, with the restrictions $Ay \leq b$, as set out in Lee et al (2015) and stacked as in Reid and Tibshirani (2015).

For the marginal screening prototype (MS) test, $\hat{y}_k = x_{j^*}$ where x_j is the j^{th} column of `x` and $j^* = \text{argmax}_{j \in C_k} |x_j^\top y|$, where C_k is the set of indices in the overall predictor set corresponding to predictors in the k^{th} group.

All tests test the null hypothesis $H_0 : \theta_{k^*} = 0$, where k^* is supplied by the user via `test.group`. Details of each are described in Reid et al (2015).

Value

A list with the following four components:

`ts` The value of the test statistic on the observed data.

`p.val` Valid p-value of the test.

`selected.col` Vector with columns selected for prototype formation in the test. If initially `NULL`, this will now contain indices of columns selected by the automatic column selection procedures of the test.

`y.hr` Matrix with hit-and-run replications of the response. If sampled selective test was not performed, this will be `NULL`.

Author(s)

Stephen Reid

References

Reid, S. and Tibshirani, R. (2015) *Sparse regression and marginal testing using cluster prototypes*. `http://arxiv.org/pdf/1503.00334v2.pdf`. *Biostatistics doi: 10.1093/biostatistics/kxv049*
Reid, S., Taylor, J. and Tibshirani, R. (2015) *A general framework for estimation and inference from clusters of features*. Available online: `http://arxiv.org/abs/1511.07839`.

See Also

`prototest.univariate`

Examples

```
require (prototest)

### generate data
set.seed (12345)
n = 100
p = 80

X = matrix (rnorm(n*p, 0, 1), ncol=p)


beta = rep(0, p)
beta[1:3] = 0.1 # three signal variables: number 1, 2, 3
signal = apply(X, 1, function(col){sum(beta*col)})
intercept = 3

y = intercept + signal + rnorm (n, 0, 1)

### treat all columns as if in same group and test for signal

# non-selective ELR test with nuisance intercept
elr = prototest.univariate (X, y, "ELR", selected.col=1:5)
# selective F test with nuisance intercept; non-sampling
f.test = prototest.univariate (X, y, "F", lambda=0.01, hr.iter=0)
print (elr)
print (f.test)

### assume variables occur in 4 equally sized groups
num.groups = 4
```

```
groups = rep (1:num.groups, each=p/num.groups)

# selective ALR test -- select columns 21-25 in 2nd group; test for signal in 1st; hit-and-run
alr = prototest.multivariate(X, y, groups, 1, "ALR", 21:25, lambda=0.005, hr.iter=20000)
# non-selective MS test -- specify first column in each group; test for signal in 1st
ms = prototest.multivariate(X, y, groups, 1, "MS", c(1,21,41,61))
print (alr)
print (ms)
```

prototest.univariate *Perform Prototype or F Tests for Significance of Groups of Predictors in the Univariate Model*

Description

Perform prototype or F tests for significance of groups of predictors in the univariate model. Choose either exact or approximate likelihood ratio prototype tests (ELR) or (ALR) or F test or marginal screening prototype test. Options for selective or non-selective tests. Further options for non-sampling or hit-and-run null reference distributions for selective tests.

Usage

```
prototest.univariate(x, y, type = c("ALR", "ELR", "MS", "F"),
selected.col = NULL, lambda, mu = NULL, sigma = 1, hr.iter = 50000,
hr.burn.in = 5000, verbose = FALSE, tol = 10^-8)
```

Arguments

x
: input matrix of dimension *n*-by-*p*, where *p* is the number of predictors in a single predetermined group of predictors. Will be mean centered and standardised before tests are performed.

y
: response variable. Vector of length emphn, assumed to be quantitative.

type
: type of test to be performed. Can only select one at a time. Options include the exact and approximate likelihood ratio prototype tests of Reid et al (2015) (ELR, ALR), the F test and the marginal screening prototype test of Reid and Tibshirani (2015) (MS). Default is ELR.

selected.col
: preselected columns specified by user. Vector of indices in the set {1, 2, ..., *p*}. If specified, a *non-selective* (classical) version of the chosen test it performed. In particular, this means the classicial χ^2_1 reference distribution for the likelihood ratio tests and the F reference for the F test. Default is `NULL`, which directs the function to estimate the selected set with the lasso or the marginal screening procedure, depending on the test.

lambda
: regularisation parameter for the lasso fit. Must be supplied when `selected.col` is `NULL`. Will be supplied to `glmnet`. This is the unstandardised version, equivalent to `lambda`/*n* supplied to `glmnet`.

`mu` mean parameter for the response. See Details below. If supplied, it is first subtracted from the response to yield a mean-zero (at the population level) vector for which we proceed with testing. If `NULL` (the default), this parameter is treated as nuisance parameter and accounted for as such in testing.

`sigma` error standard deviation for the response. See Details below. Must be supplied. If not, it is assumed to be 1. Required for the computation of some of the test statistics.

`hr.iter` number of hit-and-run samples required in the reference distrbution of a selective test. Applies only if `selected.col` is `NULL`. Default is 50000. Since dependent samples are generated, large values are required to generate good reference distributions. If set to 0, the function tries to apply a non-sampling selective test (provided `selected.col` is `NULL`), if possible. If non-sampling test is not possible, the function exits with a message.

`hr.burn.in` number of burn-in hit-and-run samples. These are generated first so as to make subsequent hit-and-run realisations less dependent on the observed response. Samples are then discarded and do not inform the null reference distribution.

`verbose` should progress be printed?

`tol` convergence threshold for iterative optimisation procedures.

Details

The model underpinning each of the tests is

$$y = \mu + \theta \cdot \hat{y} + \epsilon$$

where $\epsilon \sim N(0, \sigma^2 I)$ and $\hat{y}$ depends on the particular test considered.

In particular, for the ELR, ALR and F tests, we have $\hat{y} = P_M \left(y - \mu\right)$, where $P_M = X_M \left(X_M^\top X_M\right)^{-1} X_M^\top$. X_M is the input matrix reduced to the columns in the set *M*, which, in turn, is either provided by the user (via `selected.col`) or selected by the lasso (if `selected.col` is `NULL`). If the former, a non-selective test is performed; if the latter, a selective test is performed, with the restrictions $Ay \leq b$, as set out in Lee et al (2015).

For the marginal screening prototype (MS) test, $\hat{y} = x_{j^*}$ where x_j is the j^{th} column of `x` and $j^* = \text{argmax}_j |x_j^\top y|$.

All tests test the null hypothesis $H_0 : \theta = 0$. Details of each are described in Reid et al (2015).

Value

A list with the following four components:

`ts` The value of the test statistic on the observed data.

`p.val` Valid p-value of the test.

`selected.col` Vector with columns selected. If initially `NULL`, this will now contain indices of columns selected by the automatic column selection procedures of the test.

`y.hr` Matrix with hit-and-run replications of the response. If sampled selective test was not performed, this will be `NULL`.

Author(s)

Stephen Reid

References

Reid, S. and Tibshirani, R. (2015) *Sparse regression and marginal testing using cluster prototypes*. http://arxiv.org/pdf/1503.00334v2.pdf. *Biostatistics doi: 10.1093/biostatistics/kxv049*
Reid, S., Taylor, J. and Tibshirani, R. (2015) *A general framework for estimation and inference from clusters of features*. Available online: http://arxiv.org/abs/1511.07839.

See Also

`prototest.multivariate`

Examples

```
require (prototest)

### generate data
set.seed (12345)
n = 100
p = 80

X = matrix (rnorm(n*p, 0, 1), ncol=p)


beta = rep(0, p)
beta[1:3] = 0.1 # three signal variables: number 1, 2, 3
signal = apply(X, 1, function(col){sum(beta*col)})
intercept = 3

y = intercept + signal + rnorm (n, 0, 1)

### treat all columns as if in same group and test for signal

# non-selective ELR test with nuisance intercept
elr = prototest.univariate (X, y, "ELR", selected.col=1:5)
# selective F test with nuisance intercept; non-sampling
f.test = prototest.univariate (X, y, "F", lambda=0.01, hr.iter=0)
print (elr)
print (f.test)

### assume variables occur in 4 equally sized groups
num.groups = 4
groups = rep (1:num.groups, each=p/num.groups)

# selective ALR test -- select columns 21-25 in 2nd group; test for signal in 1st; hit-and-run
alr = prototest.multivariate(X, y, groups, 1, "ALR", 21:25, lambda=0.005, hr.iter=20000)
# non-selective MS test -- specify first column in each group; test for signal in 1st
ms = prototest.multivariate(X, y, groups, 1, "MS", c(1,21,41,61))
print (alr)
print (ms)
```

Index