Package ‘leapp’

May 8, 2026

Version 1.3

Date 2022-06-19

Title Latent Effect Adjustment After Primary Projection

Author Yunting Sun <`yunting.sun@gmail.com`> , Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

Maintainer Yunting Sun <`yunting.sun@gmail.com`>

Description These functions take a gene expression value matrix, a primary covariate vector, an additional known covariates matrix. A two stage analysis is applied to counter the effects of latent variables on the rankings of hypotheses. The estimation and adjustment of latent effects are proposed by Sun, Zhang and Owen (2011). ``leapp'' is developed in the context of microarray experiments, but may be used as a general tool for high throughput data sets where dependence may be involved.

Depends R (>= 3.1.1), sva, MASS, corpcor

License GPL (>= 2)

Repository CRAN

Date/Publication 2022-06-19 21:10:02 UTC

NeedsCompilation no

Contents

`leapp-package` *latent effect adjustment after primary projection*

Description

These functions take a gene expression value matrix, a primary covariate vector, an additional known covariates matrix. A two stage analysis is applied to counter the effects of latent variables on the rankings of hypotheses. The estimation and adjustment of latent effects are proposed by Sun, Zhang and Owen (2011). "leapp" is developed in the context of microarray experiments, but may be used as a general tool for high throughput data sets where dependence may be involved.

Details

Package:	leapp
Type:	Package
Version:	1.1
Date:	2013-01-05
License:	What license is it under?
LazyLoad:	yes

Author(s)

Maintainer: Yunting Sun <yunting.sun@gmail.com>

See Also

Sun, Zhang and Owen (2011), "Multiple hypothesis testing, adjusting for latent variables"

Examples

```
## Not run:
  library(sva)
  library(MASS)
  library(leapp)
  data(simdat)
  model <- cbind(rep(1,60),simdat$g)
  model0 <- cbind(rep(1,60))
```

```
    p.raw <- f.pvalue(simdat$data,model,model0)
    p.oracle <-f.pvalue(simdat$data - simdat$u

    p.leapp <- leapp(simdat$data,pred.prim = simdat$g)$p
    p = cbind(p.raw,p.oracle, p.leapp)
    topk = seq(0,0.5,length.out = 50)*1000
    null.set = which(simdat$gamma !=0)
    fpr= apply(p,2,FindFpr,null.set,topk)
    tpr= apply(p,2,FindTpr,null.set,topk)
    ROCplot(fpr,tpr, main = "ROC Comparison",
            name.method = c("raw","oracle","leapp"), save = FALSE )

## End(Not run)
```

`AlternateSVD` *Alternating singular value decomposition*

Description

The algorithm alternates between 1) computing latent loadings u and latent variable v and 2) estimating noise standard deviation for each of the N genes.

Usage

```
AlternateSVD(x, r, pred = NULL, max.iter = 10, TOL = 1e-04)
```

Arguments

`x`	an N by n data matrix
`r`	a numeric, number of latent factors to estimate
`pred`	an n by s matrix, each column is a vector of known covariates for n samples, s covariates are considered, default to `NULL`
`max.iter`	a numeric, maximum number of iteration allowed, default to 10
`TOL`	a numeric, tolerance level for the algorithm to converge, default to 1e-04

Value

`sigma`	a vector of length N, noise standard deviations for N genes
`coef`	an N by s matrix, estimated coefficients for known covariates
`uest`	an N by r matrix, estimated latent loadings
`vest`	an n by r matrix, estiamted latent factors

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

`FindAUC` *Compute the area under the ROC curve (AUC)*

Description

Given a vector of p values for m genes and a set of null genes, compute the area under ROC curve using the Wilcoxin statistics

Usage

```
FindAUC(pvalue, ind)
```

Arguments

`pvalue`	A vector of p values, one for each gene, with length m
`ind`	A vector of indices that the corresponding gene are true positive

Value

`auc`	A numeric, area under the ROC curve for the given gene list

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

`FindFpr` *Compute the false positive rate at given sizes of retrieved genes*

Description

Given a vector of sizes of retrieved genes, for each size k, select the top k genes with smallest p values and compute the false positive rate from the retrieved genes and the true positive genes.

Usage

```
FindFpr(pvalue, ind,topk)
```

Arguments

`pvalue`	A vector of p values, one for each gene, with length m
`ind`	A vector of indices that the corresponding gene are true positive
`topk`	A vector of integers ranging from 1 to m , length of retrieved gene list

Value

`fpr`	A vector of false positive rates at given sizes of retrieval.

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

`FindPrec` *compute the precision at given sizes of retrieved genes*

Description

Given a vector of sizes of retrieved genes, for each size k, select the top k genes with smallest p values and compute the precision from the retrieved genes and the true positive genes.

Usage

```
FindPrec(pvalue, ind,topk)
```

Arguments

`pvalue`	A vector of p values, one for each gene, with length m
`ind`	A vector of indices that the corresponding gene are true positive
`topk`	A vector of integers ranging from 1 to m , length of retrieved gene list

Value

`prec`	A vector of precisions at given sizes of retrieval.

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

`FindRec` *compute the recall at given sizes of retrieved genes*

Description

Given a vector of sizes of retrieved genes, for each size k, select the top k genes with smallest p values and compute the recall from the retrieved genes and the true positive genes.

Usage

```
FindRec(pvalue, ind, topk)
```

Arguments

`pvalue`	A vector of p values, one for each gene, with length m
`ind`	A vector of indices that the corresponding gene are true positive
`topk`	A vector of integers ranging from 1 to m , length of retrieved gene list

Value

`rec`	A vector of precisions at given sizes of retrieval.

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

`FindTpr` *compute the true positive rate at given sizes of retrieved genes*

Description

Given a vector of sizes of retrieved genes, for each size k, select the top k genes with smallest p values and compute the true positive rate from the retrieved genes and the true positive genes.

Usage

```
 FindTpr(pvalue, ind,topk)
```

Arguments

`pvalue`	A vector of p values, one for each gene, with length m
`ind`	A vector of indices that the corresponding gene are true positive
`topk`	A vector of integers ranging from 1 to m , length of retrieved gene list

Value

`tpr`	A vector of True positive rates at given sizes of retrieval.

Author(s)

Yunting Sun `<yunting.sun@gmail.com>`, Nancy R.Zhang `<nzhang@stanford.edu>`, Art B.Owen `<owen@stanford.edu>`

`IPOD` *Iterative penalized outlier detection algorithm*

Description

Outlier detection and robust regression through an iterative penalized regression with tuning parameter chosen by modified BIC

Usage

```
IPOD(X, Y, H, method = "hard", TOL = 1e-04, length.out = 50)
```

Arguments

`X`	an N by k design matrix
`Y`	an N by 1 response
`H`	an N by N projection matrix `X(X'X)^{-1}X'`
`method`	a string, if method = "hard", hard thresholding is applied; if method = "soft", soft thresholding is applied
`TOL`	relative iterative converence tolerance, default to 1e-04
`length.out`	A numeric, number of candidate tuning parameter lambda under consideration for further modified BIC model selection, default to 50.

Details

If there is no predictors, set `X = NULL`.

Y = X beta + gamma + sigma epsilon

Y is N by 1 reponse vector, X is N by k design matrix, beta is k by 1 coefficients, gamma is N by 1 outlier indicator, sigma is a scalar and the noise standard deviation and epsilon is N by 1 vector with components independently distributed as standard normal N(0,1).

Value

`gamma`	a vector of length N, estimated outlier indicator gamma
`resOpt.scale`	a vector of length N, test statistics for each of the N genes
`p`	a vector of length N, p-values for each of the N genes

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

`IPODFUN` *compute the iterative penalized outlier detection given the noise standard deviation sigma*

Description

Y = X beta + gamma + sigma epsilon estimate k by 1 coefficients vector beta and N by 1 outlier indicator vector gamma from (Y,X).

Usage

```
IPODFUN(X, Y, H, sigma, betaInit, method = "hard", TOL = 1e-04)
```

Arguments

`X`	an N by k design matrix
`Y`	an N by 1 response vector
`H`	an N by N projection matrix X(X'X)^-1X'
`sigma`	a numeric, noise standard deviation
`betaInit`	a k by 1 initial value for coeffient beta
`method`	a string, if "hard", conduct hard thresholding, if "soft", conduct soft thresholding, default to "hard"
`TOL`	a numeric, tolerance of convergence, default to 1e-04

Details

The initial estimator for the coefficient beta can be chosen to be the estimator from a robust linear regression

Value

`gamma`	an N by 1 vector of estimated outlier indicator
`ress`	an N by 1 vector of residual Y - X beta - gamma

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

References

She, Y. and Owen, A.B. "Outlier detection using nonconvex penalized regression" 2010

`leapp` *latent effect adjustment after primary projection*

Description

Adjust for latent factors and conduct multiple hypotheses testing from gene expression data using the algorithm of Sun,Zhang and Owen (2011). Number of latent factors can be chosen by Buja and Eyuboglu (1992).

Usage

```
leapp (data,pred.prim,pred.covar,
        O = NULL, num.fac = "buja", method = "hard", sparse = TRUE,
        centered = FALSE, verbose = FALSE, perm.num = 50,
        TOL = 1e-4, length.out = 50)
```

Arguments

`data`	An N genes by n arrays matrix of expression data
`pred.prim`	An n by 1 primary predictor
`pred.covar`	An n by s known covariate matrix not of primary interest
`O`	An n by n rotation matrix such that O pred.prim = (1, 0,...,0)
`num.fac`	A numeric or string, number of latent factors chosen. it has default value "buja" which uses Buja and Eyuboglu (1992) to pick the number of factors
`method`	A string which takes values in ("hard","soft"). "hard": hard thresholding in the IPOD algorithm; "soft": soft thresholding in the IPOD algorithm
`sparse`	A logical value, if `TRUE`, the signal is sparse and the proportion of non-null genes is small, use IPOD algorithm in Owen and She (2010) to enforce sparsity. If `FALSE`, the signal is not sparse, use ridge type penalty to carry out the inference as in Sun,Zhang, Owen (2011). Default to `TRUE`
`centered`	A logical value, indicates whether the data has been centered at zero, default to `FALSE`
`verbose`	A logical value, if `TRUE`, will print much information as algorithm proceeds, default to `FALSE`
`perm.num`	A numeric, number of permutation performed using algorithm of Buja and Eyuboglu (1992), default to 50
`TOL`	A numeric, convergence tolerance level, default to 1e-4
`length.out`	A numeric, number of candidate tuning parameter lambda under consideration for further modified BIC model selection, default to 50.

Details

The data for test i should be in the ith row of data. If the rotation matrix O is set to NULL, the function will compute one rotation from primary predictor pred.prim.

Value

`p`	A vector of p-values one for each row of data.
`vest`	An n by num.fac matrix, estimated latent factors
`uest`	An N by num.fac matrix, estimated latent loadings
`gamma`	An N by 1 vector, estimated primary effect
`sigma`	An N by 1 vector, estimated noise standard deviation one for each row of data

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

Examples

```
  ## Not run:
  ## Load data
  data(simdat)



  #Calculate the p-values
  p <- leapp(simdat$data,pred.prim = simdat$g,method = "hard")$p
  auc <- FindAUC(p, which(simdat$gamma!=0))



## End(Not run)
```

`Pvalue` *Calculate statistics and p-values*

Description

Calculate F-statistics, t-statistics and corresponding p-values given multiple regression models under the null and alternative hypotheses.

Usage

```
Pvalue(dat, mod, mod0)
```

Arguments

`dat`	An N genes by n arrays matrix of expression data
`mod`	An n by (s+1) design matrix under the full model (alternative), the first column is the primary predictor, s>=0 and the rest of the columns are additional covariates
`mod0`	An n by s design matrix under the null hypothesis, s>=0, should be the same as the 2:(s+1) columns of mod. If s=0, please set mod0 = `NULL`

Value

`p`	An N by 1 vector of p-values one for each row of data.
`tstat`	An N by 1 vector of t-statistics for primary parameters.
`fstat`	An N by 1 vector of F-statistics for primary parameters.
`coef`	An N by (s+1) matrix of coefficients with respect to design matrix mod under the full model.

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

Examples

```
  ## Not run:
      data(simdat)
      n = ncol(simdat$data)
      mod = cbind(simdat$g, rep(1,n))
      mod0 = cbind(rep(1,n))
      result = Pvalue(dimdat$data,mod = mod, mod0 = mod0)

## End(Not run)
```

`ridge` *Outlier detection with a ridge penalty*

Description

Outlier detection and robust regression with a ridge type penalty on the outlier indicator gamma. Allow non sparse outliers and require known noise standard deviation.

Usage

```
ridge(X, Y, H, sigma)
```

Arguments

`X`	an N by k design matrix
`Y`	an N by 1 response vector
`H`	an N by N projection matrix `X(X'X)^{-1}X'`
`sigma`	a numeric, noise standard deviation

Value

`p`	an N by 1 vector of p-values for each of the N genes
`gamma`	an N by 1 vector of estimated primary variable gamma

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

`ROCplot` *plot ROC curve*

Description

Input an p by d matrix, each column of which contains false positive rates(FPR) computed from each of the d methods and p significance levels and a matrix of corresponding true positive rates(TPR) at the same significance levels. Plot ROC curve for each of the d methods.

Usage

```
ROCplot(fpr,tpr,main, name.method,
        xlim = c(0,0.2),ylim = c(0.4,1), save = TRUE, name.file = NULL)
```

Arguments

`fpr`	A matrix of false positive rates for increasing sizes of retrieved significant genes
`tpr`	A vector of corresponding true positive rates at the same significance levels
`main`	a string, title of the figure
`name.method`	a string vector of length d containing names of the d methods
`xlim`	the range of the x axis(FPR), default to c(0,0.2)
`ylim`	the range of the y axis (TPR), default to c(0.4,1)
`save`	a logical value, if `TRUE`, will save the plot to an png file, default to `TRUE`
`name.file`	a string giving the name of the png file to save the plot

Details

The order of the name.method should be the same as that in the fpr and tpr.

Author(s)

Yunting Sun <`yunting.sun@gmail.com`>, Nancy R.Zhang <`nzhang@stanford.edu`>, Art B.Owen <`owen@stanford.edu`>

Examples

```
   ## Not run:
    library(sva)
    library(MASS)
    library(leapp)
    data(simdat)
    model <- cbind(rep(1,60),simdat$g)
    model0 <- cbind(rep(1,60))
    p.raw <- f.pvalue(simdat$data,model,model0)
    p.oracle <-f.pvalue(simdat$data - simdat$u

    p.leapp <- leapp(simdat$data,pred.prim = simdat$g, method = "hard")$p
    p = cbind(p.raw,p.oracle, p.leapp)
    topk = seq(0,0.5,length.out = 50)*1000
    null.set = which(simdat$gamma !=0)
    fpr= apply(p,2,FindFpr,null.set,topk)
    tpr= apply(p,2,FindTpr,null.set,topk)
    ROCplot(fpr,tpr, main = "ROC Comparison",
            name.method = c("raw","oracle","leapp"), save = FALSE )

## End(Not run)
```

---

simdat *Simulated gene expression data affected by a group variable and an unobserved factor*

Description

This data set is a simulated gene expression matrix with N(0,1) background noise and affected by two variables. gene expression values of 1000 genes from 60 samples are simulated. First 30 samples are from case group and last 30 samples are from control group. The primary treatment variable g affects ten percent of the genes and the latent variable affects uniformly on the genes. The correlation between primary treatment variable g and the latent variable is 0.5 and the SNR = 1, SLR = 0.5. The variances of noise across genes are distributed as inverse gamma. Also included in the data are a vector of normalized primary variable g, a vector of primary parameter gamma,a vector of latent factor v, a vector of latent loadings u and a vector of noise standard deviation for all genes sigma.

Usage

```
data(simdat)
```

Format

A list of 6 components

Value

`data`	A 1000 x 60 gene expression value matrix with genes in rows and arrays in columns
`g`	A vector of length 60, primary variable
`gamma`	A vector of length 1000, primary parameter
`v`	A vector of length 60, latent variable
`u`	A vector of length 1000, latent loadings
`sigma`	A vector of length 1000, noise standard deviation for each of the 1000 genes

Author(s)

Yunting Sun `<yunting.sun@gmail.com>`, Nancy R.Zhang `<nzhang@stanford.edu>`, Art B.Owen `<owen@stanford.edu>`

Index