Package 'pfa'

July 4, 2016

Type Package

Title Estimates False Discovery Proportion Under Arbitrary Covariance Dependence

Version 1.1

Date 2016-06-24

Author Jianqing Fan, Tracy Ke, Sydney Li and Lucy Xia

Maintainer Tracy Ke <zke@galton.uchicago.edu>

Description Estimate the false discovery proportion (FDP) by Principal Factor Approximation method with general known and unknown covariance dependence.

License GPL-2

Depends lars, POET, quantreg

Suggests MASS

Repository CRAN

Date/Publication 2016-07-04 09:16:46

NeedsCompilation no

R topics documented:

CEU *CCT8 genome-wise association data on CEU population*

Description

This data set uses 564 SNP genotype data and CCT8 gene expression data for 60 individuals from CEU population, where CEU stands for "Utah residents with ancestry from northern and western Europe".

Usage

```
CEU
```

Format

A list of two objects x and y.

- x: A matrix of dimension 60*1128. Each row corresponds to one individual. In row j, every two neighboring columns correspond to one SNP: (0,0) for "no polymorphism", (1,0) for "one nucleotide has polymorphism" and (0,1) for "both nucleotides have polymorphisms".
- y: A vector of dimension 60. Each element corresponds to the CCT8 gene expression level on one individual.

Source

http://pngu/mgh.harvard.edu/purcell/plink/res.shtml

ftp:/ftp.sanger.ac.uk/pub/genevar

References

Fan, Han and Gu (2012) "Estimating False Discovery Proportion Under Arbitrary Covariance Dependence" (with discussion) JASA.

---

pfa *Estimates False Discovery Proportion Under Arbitrary Covariance Dependence*

Description

This package contains functions for performing multiple testing and estimating the false discovery proportion (FDP). `pfa.test(X,...)` finds the false nulls in p hypotheses; `pfa.test(X,Y,...)` tests the difference of two multiple-dimensional population means; `pfa.gwas(X,Y,...)` performs the genome-wise association study(GWAS).

Usage

```
pfa.test(X,Y,tval,Sigma,reg="L2",K,e=0.05,gamma,mat_est="poet",plot="-log")
pfa.gwas(X,Y,tval,v,reg="L1",e=0.05,gamma,K,plot="-log")
```

Arguments

X,Y	In `pfa.test`, either X and Y are data matrices of two different samples, or X is a vector of test statistics and Y is missing. In `pfa.gwas`, X is the design matrix and Y contains observations of the response variable.
Sigma	the covariance matrix.
v	standard deviation of noises. By default, it is estimated by refitted cross-validation.

tval	a sequence of thresholding level for p-values. By default, tval is chosen automatically.
reg	method used to estimate factors. If reg="L1", the method is least absolute value regression; if reg="L2", the method is least-squares (with large outliers filtered out). Default is "L1".
K	number of factors. By default, given the covariance matrix, K is the smallest integer such that the sum of squares of the smallest (p-K) eigenvalues is no larger than e=0.01 times its trace.
e	a parameter used to choose the number of factors in PFA. Default value is 0.01.
gamma	a parameter used to estimate the true Null proportion: pi0 = (percentage of (p-values > gamma)) /(1- gamma). By default, it is chosen automatically.
mat_est	method used to estimate the covariance matrix. If mat_est="sample", the estimate is the (pooled) sample covariance matrix; if mat_est="poet", the estimate comes from the poet package. Default is "poet".
plot	plotting mode. If plot="-log", in the FDP plot, the x axis is -log(t); if plot="log", the x axis is log(t); if plot="linear", the x axis is t; if plot="none", no graph is generated. Default is "-log".

Details

`pfa.test(X,Sigma=Sigma,...)`: X is a vector of test statistics. Suppose it has a multivariate Gaussian distribution `N(mu, Sigma)`. We would like to test: `mu(i)=0, i=1,...,p`. Given a threshold `t`, we reject hypothesis `i` if and only if `P(i)=X(i)/sqrt(Sigma(i,i))<t,i=1,...,p`. We apply the PFA method [Fan, Han and Gu (2012)] to estimate the false discovery proportion (FDP) for arbitrary thresholds. In this case, the covariance matrix `Sigma` is required.

`pfa.test(X,...)`: X is an n-by-p matrix containing i.i.d. samples of the multivariate Gaussian distrubtion `N(mu, Sigma)`. Again, we would like to test: `mu(i)=0`, i=1,...,p. The test statistics is the vector of sample mean. When `Sigma` is unknown, we instead use the sample covariance matrix or the estimate from POET-PFA method [Fan and Han (2016)]. The number of factors determined for POET and for PFA is based on the eigenvalue ratio test in Ahn and Horenstein (2013).

`pfa.test(X,Y,...)`: X and Y are i.i.d. samples from the distributions `N(mu1, Sigma)` and `N(mu2, Sigma)`. We would like to test: `mu1(i)=mu2(i)`, i=1,...,p. The test statistics is the vector of sample mean difference. When `Sigma` is unknown, we instead use the pooled sample covariance matrix or the estimate from POET-PFA method.

`pfa.gwas(X,Y,...)`: X is the data matrix of `p` covariates (e.g. SNP measurements) and Y is the data vector of response variables (e.g. indicator of a trait). We would like to test: whether covariate `i` is marginally associated with the response, `i=1,...,p`. The test statistics is the maginal regression coefficients. We suppose `Y=X_1*beta_1+...+X_p*beta_p+epsilon`, where `epsilon`'s are i.i.d. samples from `N(0,sigma^2)`. Then the covariane matrix of the test statistics is nothing but the sample covariance matrix of X.

Value

Four graphs will be generated for each of the functions: one histogram of p-values; number of total rejections, number of false rejections, and FDPs all indexed by t.

It returns an object of class `PDFresults`, which is a list containing the following components:

Pvalue	Sorted p-values.
adjPvalue	Sorted adjusted p-values.
FDP	Estimated FDPs.
pi0	Estimated true null proportion.
K	Number of factors used in the PFA method.
sigma	Estimated standard deviation of noises. This component is NULL except in the return of pfaRegress.

Note

1. The estimated FDP does not necessarily decrease as the threshold t increases (although the number of total rejections and estimated false rejections both decrease as t increases). As a result, the estimated FDP curve is sometimes zigzag. Moreover, two values of t can yield to different estimated FDP values even if they give exactly the same rejections.

2. In pfa.gwas, when the standard deviation of noises, v, is not provided, we apply refitted cross validation [Fan,Guo and Hao (2012)] to estimate v. This may take some time (especially when the dimension p is large), and the results can be different at each running, due to random data splits. An input of v is suggested in this case.

3. Sometimes people want to use the sequence of obtained p-values as the sequence of thresholds. This can be implemented by setting tval="pval", see Example 4 below.

4. It is generally better to use the factor-adjusted p-values, but there is no universal conclusion on which is better. One way is to plot both histograms and see, if ignoring a neighborhood of 0, which one is closer to the uniform distribution.

Author(s)

Jianqing Fan, Tracy Ke, Sydney Li and Lucy Xia.

Maintainer: Tracy Ke <zke@galton.uchicago.edu>, Lucy Xia <lucyxia@stanford.edu>.

References

Fan, Han and Gu (2012) "Estimating False Discovery Proportion Under Arbitrary Covariance Dependence" (with discussion) JASA.

Fan and Han (2016) "Estimation of False Discovery Proportion with Unknown Dependence", Manuscript.

See Also

Ahn and Horestein (2013) "Eigenvalue Ratio Test for the Number of Factors", Econometrica.

Fan, Guo and Hao (2012)"Variance Estimation Using Refitted Cross-Validation in Ultrahigh Dimensional Regression" JRSSB.

Fan, Liao and Mincheva (2013) "Large Covariance Estimation by Thresholding Principal Orthogonal Complements", JRSSB .

Examples

```
# Example 1: multiple testing with known covariance

require(MASS)
p <- 100
Sigma <- matrix(0.4,p,p)
diag(Sigma)<- 1
mu <- as.vector(c(rep(3,5), rep(0, p-5)))
Z <- mvrnorm(1, mu, Sigma)
RE1 <- pfa.test(Z, Sigma=Sigma,reg="L1")
summary(RE1)

# Example 2: multiple testing with unknown covariance

n <- 200
p <- 300
K <- 3
mu <- as.vector(c(rep(2,10),rep(0,p-10)))
B <- matrix(runif(K*p, min=-1, max=1), nrow=K)
f <- matrix(rnorm(K*n), nrow=n)
Bf <- f %*% B
X <- matrix(rep(0, n*p), nrow=n)
for (i in 1:n)
   X[i,] <- mu + Bf[i,] + rnorm(p)
## Not run: RE2 <- pfa.test(X, tval="pval")
## Not run: summary(RE2)

# Example 3: testing the marginal regression coefficients

n <- 100
p <- 300
beta <- as.matrix(c(rep(2, 10), rep(0, p-10)))
X <- matrix(rep(0, n*p), nrow=n)
X[,1:10] <- matrix(rnorm(n*10), nrow=n)
z <- as.matrix(rnorm(n))
y <- as.matrix(rnorm(n))
X[,11:p] <- as.matrix(rnorm(n*(p-10)), nrow=n)
for (i in 11:p) {
    rho1 <- runif(1,min=-0.2,max=0.2)
    rho2 <- runif(1,min=-0.2,max=0.2)
    X[,i] <- X[,i] + z*rho1 + y*rho2
}
eps <- as.matrix(rnorm(n))
Y <- X %*% beta + eps
## Not run: RE3 <- pfa.gwas(X,Y)
## Not run: summary(RE3)

# Example 4: GWAS on the CCT8 gene

data(CEU)
## Not run: RE4 <- pfa.gwas(CEU$x, CEU$y, t=exp(-seq(1.8,3.6,0.1)), reg="L2")
## Not run: summary(RE4)
```

Index