Package 'BCSub'

January 20, 2025

Type Package

Title A Bayesian Semiparametric Factor Analysis Model for Subtype Identification (Clustering)

Version 0.5

Date 2017-03-16

Author Jiehuan Sun [aut, cre], Joshua L. Warren [aut], and Hongyu Zhao [aut]

Maintainer Jiehuan Sun <jiehuan.sun@yale.edu>

Description Gene expression profiles are commonly utilized to infer disease subtypes and many clustering methods can be adopted for this task. However, existing clustering methods may not perform well when genes are highly correlated and many uninformative genes are included for clustering. To deal with these challenges, we develop a novel clustering method in the Bayesian setting. This method, called BCSub, adopts an innovative semiparametric Bayesian factor analysis model to reduce the dimension of the data to a few factor scores for clustering. Specifically, the factor scores are assumed to follow the Dirichlet process mixture model in order to induce clustering.

License GPL-2

LazyData TRUE

Depends R (>= 3.0), MASS (>= 7.3-45), mcclust (>= 1.0), nFactors (>= 2.3.3)

Imports Rcpp (>= 0.12.6)

Suggests knitr

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 5.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-03-16 23:41:02 UTC

Contents

BCSub *A Bayesian semiparametric factor analysis model for subtype identification (Clustering).*

Description

A Bayesian semiparametric factor analysis model for subtype identification (Clustering).

Usage

```
BCSub(A = NULL, iter = 1000, seq = 200:1000, M = 5)
```

Arguments

A	Data matrix with rows being subjects and columns being genes.
iter	Total number of iterations (including burn-in period).
seq	Posterior samples used for inference of cluster structure.
M	Number of factors.

Value

returns a list with following objects.

CL	Inferred cluster strucutre based on the posterior samples.
E	A matrix with each column being the cluster structre at each iteration.

References

A Bayesian Semiparametric Factor Analysis Model for Subtype Identification. Jiehuan Sun, Joshua L. Warren, and Hongyu Zhao.

Examples

```
set.seed(1)
n = 100 ## number of subjects
G = 200 ## number of genes
SNR = 0 ## ratio of noise genes
## loading matrix with four factors
lam = matrix(0,G,4)
lam[1:(G/4),1] = runif(G/4,-3,3)
lam[(G/4+1):(G/2),2] = runif(G/4,-3,3)
lam[(G/2+1):(3*G/4),3] = runif(G/4,-3,3)
```

```
lam[(3*G/4+1):(G),4] = runif(G/4,-3,3)
## generate low-rank covariance matrix
sigma <- lam%*%t(lam) + diag(rep(1,G))
sigma <- cov2cor(sigma)
## true cluster structure ##
e.true = c(rep(1,n/2),rep(2,n/2))

## generate data matrix ##
mu1 = rep(1,G)
mu1[sample(1:G,SNR*G)] = 0
mu2 <- rep(0,G)
A = rbind(mvrnorm(n/2,mu1,sigma),mvrnorm(n/2,mu2,sigma))

## factor analysis to decide the number of factors
## Not run:
ev = eigen(cor(A))
ap = parallel(subject=nrow(A),var=ncol(A),rep=100,cent=.05)
nS = nScree(x=ev$values, aparallel=ap$eigen$qevpea)
M = nS$Components[1,3] ## number of factors

## End(Not run)
M = 4
## run BCSub for clustering
iters = 1000 ## total number of iterations
seq = 600:1000 ## posterior samples used for inference
system.time(res <- BCSub(A,iter=iters,seq=seq,M=M))
res$CL ## inferred cluster structure

## calculate and plot similarity matrix
sim = calSim(t(res$E[,seq]))

## plot similarity matrix
x <- rep(1:n,times=n)
y <- rep(1:n,each=n)
z <- as.vector(sim)
levelplot(z~x*y,col.regions=rev(gray.colors(n^2)), xlab = "Subject ID",ylab = "Subject ID")
```

---

calSim *Function to calculate the similarity matrix based on the cluster membership indicator of each iteration.*

Description

Function to calculate the similarity matrix based on the cluster membership indicator of each iteration.

Usage

```
calSim(mat)
```

Arguments

mat	A matrix of cluster membership indicators.

Value

returns a similarity matrix.

Examples

```
n = 90 ## number of subjects
iters = 200 ## number of iterations
## matrix of cluster membership indicators
## perfect clustering with three clusters
mat = matrix(rep(1:3,each=n/3),nrow=n,ncol=iters)
sim = calSim(t(mat))
## plot similarity matrix
x <- rep(1:n,times=n)
y <- rep(1:n,each=n)
z <- as.vector(sim)
levelplot(z~x*y,col.regions=rev(gray.colors(n^2)), xlab = "Subject ID",ylab = "Subject ID")
```

Index