Package 'debCAM'

May 5, 2024

Type Package

Title Deconvolution by Convex Analysis of Mixtures

Version 1.22.0

Author Lulu Chen <luluchen@vt.edu>

Maintainer Lulu Chen <luluchen@vt.edu>

biocViews Software, CellBiology, GeneExpression

Description An R package for fully unsupervised deconvolution of complex tissues. It provides basic functions to perform unsupervised deconvolution on mixture expression profiles by Convex Analysis of Mixtures (CAM) and some auxiliary functions to help understand the subpopulation-specific results. It also implements functions to perform supervised deconvolution based on prior knowledge of molecular markers, S matrix or A matrix. Combining molecular markers from CAM and from prior knowledge can achieve semi-supervised deconvolution of mixtures.

License GPL-2

Encoding UTF-8

RoxygenNote 6.1.1

Depends R (>= 3.5)

Suggests knitr, rmarkdown, BiocStyle, testthat, GEOquery, rgl

VignetteBuilder knitr

Imports methods, rJava, BiocParallel, stats, Biobase, SummarizedExperiment, corpcor, geometry, NMF, nnls, DMwR2, pcaPP, apcluster, graphics

SystemRequirements Java (>= 1.8)

BugReports https://github.com/Lululuella/debCAM/issues

git_url https://git.bioconductor.org/packages/debCAM

git_branch RELEASE_3_19

git_last_commit b78394e

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-05

Contents

`debCAM-package` *debCAM: A package for fully unsupervised deconvolution of complex tissues.*

Description

The core function in this package is `CAM` which achieves fully unsupervised deconvolution on mixture expression profiles. Each step in `CAM` can also be performed separately by `CAMPrep`, `CAMMGCluster` and `CAMASest` in a more flexible workflow. `MGstatistic` can help extract a complete marker list from CAM results. `MDL` can help decide the underlying subpopulation number. With other functions, e.g. `AfromMarkers` and `MGstatistic`, this package can also perform supervised deconvolution based on prior knowledge of molecular markers, subpopulation-specific expression matrix (S) or proportion matrix (A). Semi-supervised deconvolution can be achieved by combining molecular markers from CAM and from prior knowledge to analyze mixture expressions.

References

Wang, N., Hoffman, E. P., Chen, L., Chen, L., Zhang, Z., Liu, C., ... Wang, Y. (2016). Mathematical modelling of transcriptional heterogeneity identifies novel markers and subpopulations in complex tissues. Scientific Reports, 6, 18909. http://doi.org/10.1038/srep18909

`AfromMarkers` *Proportion matrix estimation from marker genes*

Description

This function estimates proportion matrix (A matrix) from observed mixture expression data based on marker genes.

Usage

```
AfromMarkers(data, MGlist, scaleRecover = TRUE)
```

Arguments

`data`	A data set that will be internally coerced into a matrix. Each row is a gene and each column is a sample. data should be in non-log linear space with non-negative numerical values (i.e. >= 0). Missing values are not supported. All-zero rows will be removed internally.
`MGlist`	A list of vectors, each of which contains known markers and/or CAM-detected markers for one subpopulation.
`scaleRecover`	If TRUE, scale ambiguity of each column vector in A matrix is removed based on sum-to-one constraint on each row.

Details

With the expression levels of subpopulation-specific marker genes, the relative proportions of constituent subpopulations are estimated by spatial median using `l1median`. Marker genes could be from unsupervised/supervised detection or from literatures. Scale ambiguity is optionally removed based on sum-to-one constraint of rows.

Value

Return the estimated proportion matrix (A matrix).

Examples

```
#obtain data and marker genes
data(ratMix3)
S <- ratMix3$S
pMGstat <- MGstatistic(S, c("Liver","Brain","Lung"))
pMGlist.FC <- lapply(c("Liver","Brain","Lung"), function(x)
    rownames(pMGstat)[pMGstat$idx == x & pMGstat$OVE.FC > 10])
```

```
#estimate A matrix from markers
Aest <- AfromMarkers(ratMix3$X, pMGlist.FC)
```

AS-accessor *Deconvoluted matrix accessors*

Description

Accessors to proportion matrix and subpopulation-specific expression matrix estimated by CAM.

Usage

```
Amat(x, ...)

Smat(x, ...)

## S4 method for signature 'CAMObj'
Amat(x, k, usingPCA = TRUE)

## S4 method for signature 'CAMASObj'
Amat(x, usingPCA = TRUE)

## S4 method for signature 'CAMObj'
Smat(x, k, usingPCA = TRUE)

## S4 method for signature 'CAMASObj'
Smat(x, usingPCA = TRUE)
```

Arguments

x	a CAMObj object or a CAMASObj object.
...	additional argument list.
k	subpopulation number
usingPCA	If TRUE, A matrix is estimated by transforming dimension-reduced A matrix back to original space. Otherwise, A matrix is directly estimated in original data space. The default is TRUE.

Value

Estimated A matrix or S matrix.

Examples

```
#obtain data
data(ratMix3)
data <- ratMix3$X

rCAM <- CAM(data, K = 3, dim.rdc = 3, thres.low = 0.30, thres.high = 0.95)
Aest <- Amat(rCAM, 3)
Sest <- Smat(rCAM, 3)

Aest <- Amat(slot(rCAM, "ASestResult")[[1]])
Sest <- Smat(slot(rCAM, "ASestResult")[[1]])
```

CAM *Convex Analysis of Mixtures*

Description

This function performs a fully unsupervised computational deconvolution to identify marker genes that define each of the multiple subpopulations, and estimate the proportions of these subpopulations in the mixture tissues as well as their respective expression profiles.

Usage

```
CAM(data, K = NULL, corner.strategy = 2, dim.rdc = 10,
  thres.low = 0.05, thres.high = 0.95, cluster.method = c("K-Means",
  "apcluster"), cluster.num = 50, MG.num.thres = 20,
  lof.thres = 0.02, quickhull = TRUE, quick.select = NULL,
  sample.weight = NULL, appro3 = TRUE, generalNMF = FALSE,
  cores = NULL)
```

Arguments

data	Matrix of mixture expression profiles. Data frame, SummarizedExperiment or ExpressionSet object will be internally coerced into a matrix. Each row is a gene and each column is a sample. Data should be in non-log linear space with non-negative numerical values (i.e. >= 0). Missing values are not supported. All-zero rows will be removed internally.
K	The candidate subpopulation number(s), e.g. K = 2:8.
corner.strategy	The method to find corners of convex hull. 1: minimum sum of margin-of-errors; 2: minimum sum of reconstruction errors. The default is 2.
dim.rdc	Reduced data dimension; should be not less than maximum candidate K.
thres.low	The lower bound of percentage of genes to keep for CAM with ranked norm. The value should be between 0 and 1. The default is 0.05.
thres.high	The higher bound of percentage of genes to keep for CAM with ranked norm. The value should be between 0 and 1. The default is 0.95.

`cluster.method` The method to do clustering. The default "K-Means" will use `kmeans`. The alternative "apcluster" will use `apclusterK-methods`.

`cluster.num` The number of clusters; should be much larger than K. The default is 50.

`MG.num.thres` The clusters with the gene number smaller than MG.num.thres will be treated as outliers. The default is 20.

`lof.thres` Remove local outlier using `lofactor`. MG.num.thres is used as the number of neighbors in the calculation of the local outlier factors. The default value 0.02 will remove top 2% local outliers. Zero value will disable lof.

`quickhull` Perform quickhull to select clusters or not. The default is True.

`quick.select` The number of candidate corners kept after quickhull and SFFS greedy search. If Null, only quickhull is applied. The default is 20. If this value is larger than the number of candidate corners after quickhull, greedy search will also not be applied.

`sample.weight` Vector of sample weights. If NULL, all samples have the same weights. The length should be the same as sample numbers. All values should be positive.

`appro3` Estimate A and S matrix by approach 3 or not. Please see `CAMASest` for further information. The default is TRUE.

`generalNMF` If TRUE, the decomposed proportion matrix has no sum-to-one constraint for each row. The default is FALSE. TRUE value brings two changes: (1) Without assuming samples are normalized, the first principal component will not forced to be along c(1,1,..,1) but a standard PCA will be applied during preprocessing. (2) Without sum-to-one constraint for each row, the scale ambiguity of each column vector in proportion matrix will not be removed.

`cores` The number of system cores for parallel computing. If not provided, one core for each element in K will be invoked. Zero value will disable parallel computing.

Details

This function includes three necessary steps to decompose a matrix of mixture expression profiles: data preprocessing, marker gene cluster search, and matrix decomposition. They are implemented in `CAMPrep`, `CAMMGCluster` and `CAMASest`, separately. More details can be found in the help document of each function.

For this function, you needs to specify the range of possible subpopulation numbers and the percentage of low/high-expressed genes to be removed. Typically, 30% ~ 50% low-expressed genes can be removed from gene expression data. The removal of high-expressed genes has much less impact on results, and usually set to be 0% ~ 10%.

This function can also analyze other molecular expression data, such as proteomics data. Much less low-expressed proteins need to be removed, e.g. 0% ~ 10%, due to a limited number of proteins without missing values.

Value

An object of class "`CAMObj`" containing the following components:

`PrepResult` An object of class "`CAMPrepObj`" containing data preprocessing results from `CAMPrep` function.

MGResult — A list of "CAMMGObj" objects containing marker gene detection results from CAMMGCluster function for each K value.

ASestResult — A list of "CAMASObj" objects containing estimated proportions, subpopulation-specific expressions and mdl values from CAMASest function for each K value.

Examples

```
#obtain data
data(ratMix3)
data <- ratMix3$X

#set seed to generate reproducible results
set.seed(111)

#CAM with known subpopulation number
rCAM <- CAM(data, K = 3, dim.rdc = 3, thres.low = 0.30, thres.high = 0.95)
#Larger dim.rdc can improve performance but increase time complexity

## Not run:
#CAM with a range of subpopulation number
rCAM <- CAM(data, K = 2:5, dim.rdc = 10, thres.low = 0.30, thres.high = 0.95)

#Use "apcluster" to aggregate gene vectors in CAM
rCAM <- CAM(data, K = 2:5, dim.rdc = 10, thres.low = 0.30, thres.high = 0.95,
            cluster.method = 'apcluster')

#CAM with quick selection to reduce time complexity
rCAM <- CAM(data, K = 3, dim.rdc = 10, thres.low = 0.30, thres.high = 0.95,
        quick.select = 20)

#CAM with different sample weights (e.g. adjusted based on sample quality)
rCAM <- CAM(data, K = 3, dim.rdc = 5, thres.low = 0.30, thres.high = 0.95,
        sample.weight = c(rep(10,11),rep(1,10)))

#CAM for general NMF (no sum-to-one contraint for proportion matrix)
rCAM <- CAM(data, K = 3, dim.rdc = 5, thres.low = 0.30, thres.high = 0.95,
       generalNMF = TRUE)

## End(Not run)
```

CAMASest *A and S matrix estimation by CAM*

Description

This function estimates A and S matrix based on marker gene clusters detected by CAM.

Usage

```
CAMASest(MGResult, PrepResult, data, corner.strategy = 2,
  appro3 = TRUE, generalNMF = FALSE)
```

Arguments

MGResult	An object of class "CAMMGObj" obtained from CAMMGCluster function.
PrepResult	An object of class "CAMPrepObj" obtained from CAMPrep function.
data	Matrix of mixture expression profiles which need to be the same as the input of CAMPrep. Data frame, SummarizedExperiment or ExpressionSet object will be internally coerced into a matrix. Each row is a gene and each column is a sample. Data should be in non-log linear space with non-negative numerical values (i.e. >= 0). Missing values are not supported. All-zero rows will be removed internally.
corner.strategy	The method to detect corner clusters. 1: minimum sum of margin-of-errors; 2: minimum sum of reconstruction errors. The default is 2.
appro3	Estimate A and S matrix by approach 3 or not. Please see details for further information. The default is TRUE.
generalNMF	If TRUE, the decomposed proportion matrix has no sum-to-one constraint for each row. Without this constraint, the scale ambiguity of each column vector in proportion matrix will not be removed. The default is FALSE.

Details

This function is used internally by CAM function to estimate proportion matrix (A), subpopulation-specific expression matrix (S) and mdl values. It can also be used when you want to perform CAM step by step.

The mdl values are calculated in three approaches: (1) based on data and A matrix in dimension-reduced space; (2) based on original data with A matrix estimated by transforming dimension-reduced A matrix back to original space; (3) based on original data with A directly estimated in original space. A and S matrix in original space estimated from the latter two approaches are returned. mdl is the sum of two terms: code length of data under the model and code length of model. Both mdl value and the first term (code length of data) will be returned.

Value

An object of class "CAMASObj" containing the following components:

Aest	Estimated proportion matrix from Approach 2.
Sest	Estimated subpopulation-specific expression matrix from Approach 2.
Aest.proj	Estimated proportion matrix from Approach 2, before removing scale ambiguity.
Ascale	The estimated scales to remove scale ambiguity of each column vector in Aest. Sum-to-one constraint on each row of Aest is used for scale estimation.
AestO	Estimated proportion matrix from Approach 3.
SestO	Estimated subpopulation-specific expression matrix from Approach 3.
AestO.proj	Estimated proportion matrix from Approach 3, before removing scale ambiguity.

AscaleO	The estimated scales to remove scale ambiguity of each column vector in AestO. Sum-to-one constraint on each row of AestO is used for scale estimation.
datalength	Three values for code length of data. The first is calculated based on dimension-reduced data. The second and third are based on the original data.
mdl	Three mdl values. The first is calculated based on dimension-reduced data. The second and third are based on the original data.

Examples

```
#obtain data
data(ratMix3)
data <- ratMix3$X

#preprocess data
rPrep <- CAMPrep(data, dim.rdc = 3, thres.low = 0.30, thres.high = 0.95)

#Marker gene cluster detection with a fixed K
rMGC <- CAMMGCluster(3, rPrep)

#A and S matrix estimation
rASest <- CAMASest(rMGC, rPrep, data)
```

CAMASObj-class *Class "CAMASObj"*

Description

An S4 class for storing estimated proportions, subpopulation-specific expressions and mdl values. The mdl values are calculated in three approaches: (1) based on data and A matrix in dimension-reduced space; (2) based on original data with A matrix estimated by transforming dimension-reduced A matrix back to original space; (3) based on original data with A directly estimated in original space. A and S matrix in original space estimated from the latter two approaches are returned. mdl is the sum of two terms: code length of data under the model and code length of model. Both mdl value and the first term (code length of data) will be returned.

Slots

`Aest` Estimated proportion matrix from Approach 2.

`Sest` Estimated subpopulation-specific expression matrix from Approach 2.

`Aest.proj` Estimated proportion matrix from Approach 2, before removing scale ambiguity.

`Ascale` The estimated scales to remove scale ambiguity of each column vector in Aest. Sum-to-one constraint on each row of Aest is used for scale estimation.

`AestO` Estimated proportion matrix from Approach 3.

`SestO` Estimated subpopulation-specific expression matrix from Approach 3.

`AestO.proj` Estimated proportion matrix from Approach 3, before removing scale ambiguity.

`AscaleO` The estimated scales to remove scale ambiguity of each column vector in AestO. Sum-to-one constraint on each row of AestO is used for scale estimation.

`datalength` Three values for code length of data. The first is calculated based on dimension-reduced data. The second and third are based on the original data.

`mdl` Three mdl values. The first is calculated based on dimension-reduced data. The second and third are based on the original data.

`CAMMGCluster` *MG cluster detection for CAM*

Description

This function finds corner clusters as MG clusters (clusters containing marker genes).

Usage

```
CAMMGCluster(K, PrepResult, generalNMF = FALSE, nComb = 200)
```

Arguments

`K`	The candidate subpopulation number.
`PrepResult`	An object of class "`CAMPrepObj`" obtained from `CAMPrep` function.
`generalNMF`	If TRUE, the decomposed proportion matrix has no sum-to-one constraint for each row. Without this constraint, the scale ambiguity of corner cluster centers will not be removed when computing reconstruction errors. The default is FALSE.
`nComb`	The number of possible combinations of clusters as corner clusters. Within these possible combinations ranked by margin errors, we can further select the best one based on reconstruction errors. The default is 200.

Details

This function is used internally by `CAM` function to detect clusters containing marker genes, or used when you want to perform CAM step by step.

This function provides two solutions. The first is the combination of clusters yielding the minimum sum of margin-of-errors for cluster centers. In the second, nComb possible combinations are selected by ranking sum of margin-of-errors for cluster centers. Then the best one is selected based on reconstruction errors of all data points in original space.

Value

An object of class "`CAMMGObj`" containing the following components:

`idx`	Two numbers which are two solutions' ranks by sum of margin-of-error.
`corner`	The indexes of clusters as detected corners. Each row is a solution.
`error`	Two rows. The first row is sum of margin-of-errors for nComb possible combinations. The second row is reconstruction errors for nComb possible combinations.

Examples

```
#obtain data
data(ratMix3)
data <- ratMix3$X

#preprocess data
rPrep <- CAMPrep(data, dim.rdc = 3, thres.low = 0.30, thres.high = 0.95)

#Marker gene cluster detection with a fixed K = 3
rMGC <- CAMMGCluster(3, rPrep)
```

CAMMGObj-class *Class "CAMMGObj"*

Description

An S4 class for storing marker gene detection results.

Slots

`idx` Two numbers which are two solutions' ranks by sum of margin-of-error.

`corner` The indexes of clusters as detected corners. Each row is a solution.

`error` Two rows. The first row is sum of margin-of-errors for nComb possible combinations. The second row is reconstruction errors for nComb possible combinations.

CAMObj-class *Class "CAMObj"*

Description

An S4 class for storing results of CAM.

Slots

`PrepResult` An object of class "`CAMPrepObj`" storing data preprocessing results from `CAMPrep` function.

`MGResult` A list of "`CAMMGObj`" objects storing marker gene detection results from `CAMMGCluster` function for each candidate subpopulation number.

`ASestResult` A list of "`CAMASObj`" objects storing estimated proportions, subpopulation-specific expressions and mdl values from `CAMASest` function for each candidate subpopulation number.

CAMPrep *Data preprocessing for CAM*

Description

This function perform preprocessing for CAM, including norm-based filtering, dimension deduction, perspective projection, local outlier removal and aggregation of gene expression vectors by clustering.

Usage

```
CAMPrep(data, dim.rdc = 10, thres.low = 0.05, thres.high = 0.95,
  cluster.method = c("K-Means", "apcluster"), cluster.num = 50,
  MG.num.thres = 20, lof.thres = 0.02, quickhull = TRUE,
  quick.select = NULL, sample.weight = NULL, generalNMF = FALSE)
```

Arguments

data	Matrix of mixture expression profiles. Data frame, SummarizedExperiment or ExpressionSet object will be internally coerced into a matrix. Each row is a gene and each column is a sample. Data should be in non-log linear space with non-negative numerical values (i.e. >= 0). Missing values are not supported. All-zero rows will be removed internally.
dim.rdc	Reduced data dimension; should be not less than maximum candidate K.
thres.low	The lower bound of percentage of genes to keep for CAM with ranked norm. The value should be between 0 and 1. The default is 0.05.
thres.high	The higher bound of percentage of genes to keep for CAM with ranked norm. The value should be between 0 and 1. The default is 0.95.
cluster.method	The method to do clustering. The default "K-Means" will use kmeans function. The alternative "apcluster" will use apclusterK-methods.
cluster.num	The number of clusters; should be much larger than K. The default is 50.
MG.num.thres	The clusters with the gene number smaller than MG.num.thres will be treated as outliers. The default is 20.
lof.thres	Remove local outlier using lofactor function. MG.num.thres is used as the number of neighbors in the calculation of the local outlier factors. The default value 0.02 will remove top 2% local outliers. Zero value will disable lof.
quickhull	Perform quickhull to select clusters or not. The default is True.
quick.select	The number of candidate corners kept after quickhull and SFFS greedy search. If Null, only quickhull is applied. The default is 20. If this value is larger than the number of candidate corners after quickhull, greedy search will also not be applied.
sample.weight	Vector of sample weights. If NULL, all samples have the same weights. The length should be the same as sample numbers. All values should be positive.

`generalNMF` If TRUE, the decomposed proportion matrix has no sum-to-one constraint for each row. Without assuming samples are normalized, the first principal component will not forced to be along c(1,1,..,1) but a standard PCA will be applied during preprocessing.

Details

This function is used internally by `CAM` function to preprocess data, or used when you want to perform CAM step by step.

Low/high-expressed genes are filtered by their L2-norm ranks. Dimension reduction is slightly different from PCA. The first loading vector is forced to be c(1,1,...,1) with unit norm normalization. The remaining are eigenvectors from PCA in the space orthogonal to the first vector. Perspective projection is to project dimension-reduced gene expression vectors to the hyperplane orthogonal to c(1,0,...,0), i.e., the first axis in the new coordinate system. local outlier removal is optional to exclude outliers in simplex formed after perspective projection. Finally, gene expression vectors are aggregated by clustering to further reduce the impact of noise/outlier and help improve the efficiency of simplex corner detection.

Value

An object of class "`CAMPrepObj`" containing the following components:

`Valid` logical vector to indicate the genes left after filtering.

`Xprep` Preprocessed data matrix.

`Xproj` Preprocessed data matrix after perspective projection.

`W` The matrix whose rows are loading vectors.

`SW` Sample weights.

`cluster` cluster results including two vectors. The first indicates the cluster to which each gene is allocated. The second is the number of genes in each cluster.

`c.outlier` The clusters with the gene number smaller than MG.num.thres.

`centers` The centers of candidate corner clusters (candidate clusters containing marker genes).

Examples

```
#obtain data
data(ratMix3)
data <- ratMix3$X

#set seed to generate reproducible results
set.seed(111)

#preprocess data
rPrep <- CAMPrep(data, dim.rdc = 3, thres.low = 0.30, thres.high = 0.95)
```

CAMPrepObj-class *Class "CAMPrepObj"*

Description

An S4 class for storing data preprocessing results.

Slots

`Valid` logical vector to indicate the genes left after filtering.

`Xprep` Preprocessed data matrix.

`Xproj` Preprocessed data matrix after perspective projection.

`W` The matrix whose rows are loading vectors.

`cluster` cluster results including two vectors. The first indicates the cluster to which each gene is allocated. The second is the number of genes in each cluster.

`c.outlier` The clusters with the gene number smaller than MG.num.thres.

`centers` The centers of candidate corner clusters (candidate clusters containing marker genes).

cornerSort *Candidate combinations as corners*

Description

Given a set of data points, return possible combinations of data points as corners. These combinations are selected by ranking the sum of margin-of-errors.

Usage

```
cornerSort(X, K, nComb)
```

Arguments

`X`	Data in a matrix. Each column is a data point.
`K`	The number of corner points.
`nComb`	The number of returned combinations of data points as corners. All combinations will be returned if the number of all combinations is less than nComb.

Details

This function is to detect K corner points from M data points by conducting an exhaustive combinatorial search (with total C_M^K combinations), based on a convex-hull-to-data fitting criterion: sum of margin-of-errors. `nComb` combinations are returned for further selection based on reconstruction errors of all data points in original space.

The function is implemented in Java with R-to-Java interface provided by `rJava` package. It relies on NonNegativeLeastSquares class in Parallel Java Library (https://www.cs.rit.edu/~ark/pj.shtml).

Value

A list containing the following components:

idx	A matrix to show the indexes of data points in combinations to construct a convex hull. Each column is one combination.
error	A vector of margin-of-error sums for each combination.

Examples

```
data <- matrix(c(0.1,0.2,1.0,0.0,0.0,0.5,0.3,
                 0.1,0.7,0.0,1.0,0.0,0.0,0.5,0.3,
                 0.8,0.1,0.0,0.0,0.0,1.0,0.0,0.0,0.4), nrow =3, byrow = TRUE)
topconv <- cornerSort(data, 3, 10)
```

MDL *Minimum Description Length*

Description

This function obtains minimum description length (mdl) values for each candidate subpopulation number.

Usage

```
MDL(CAMResult, mdl.method = 3)

## S4 method for signature 'MDLObj,missing'
plot(x, data.term = FALSE, ...)
```

Arguments

CAMResult	Result from CAM function.
mdl.method	Approach to calculate mdl values; should be 1, 2, or 3. The default is 3.
x	An object of class "MDLObj" from MDL.
data.term	If true, plot data term (code length of data under model).
...	All other arguments are passed to the plotting command.

Details

This function extracts minimum description length (mdl) values from the result of CAM function, which contains mdl values form three approaches for each candidate subpopulation number. For more details about three approaches, refer to CAMASest.

mdl is code length of data under the model plus code length of model. Both mdl value and the first term about data are returned.

Value

An object of class "MDLObj" containing the following components:

K	The candidate subpopulation numbers.
datalengths	For each model with a certain subpopulation number, code length of data under the model.
mdls	mdl value for each model with a certain subpopulation number.

Examples

```
## Not run:
#obtain data
data(ratMix3)
data <- ratMix3$X

#Analysis by CAM
rCAM <- CAM(data, K = 2:5, thres.low = 0.30, thres.high = 0.95)

#extract mdl values
MDL(rCAM)
MDL(rCAM, 1)
MDL(rCAM, 2)

#plot MDL curves
plot(MDL(rCAM))
plot(MDL(rCAM), data.term = TRUE) #with data length curve

## End(Not run)
```

MDLObj-class *Class "MDLObj"*

Description

An S4 class for storing mdl values.

Slots

K The candidate subpopulation numbers.

datalengths For each model with a certain subpopulation number, code length of data under the model.

mdls mdl value for each model with a certain subpopulation number.

MGsforA *Marker genes detected by CAM for estimating A*

Description

This function returns marker genes detected by CAM for estimating A.

Usage

```
MGsforA(CAMResult = NULL, K = NULL, PrepResult = NULL,
  MGResult = NULL, corner.strategy = 2)
```

Arguments

CAMResult	Result from CAM.
K	The candidate subpopulation number.
PrepResult	An object of class "CAMPrepObj" from CAMPrep.
MGResult	An object of class "CAMMGObj" from CAMMGCluster.
corner.strategy	The method to detect corner clusters. 1: minimum sum of margin-of-errors; 2: minimum sum of reconstruction errors. The default is 2.

Details

This function needs to specify CAMResult and K, or PrepResult and MGResult. The returned marker genes are those used by CAM for estimating A. To obtain a more complete marker gene list, please refer to MGstatistic.

Value

A list of vectors, each of which contains marker genes for one subpopulation.

Examples

```
#obtain data and run CAM
data(ratMix3)
data <- ratMix3$X
rCAM <- CAM(data, K = 3, dim.rdc= 3, thres.low = 0.30, thres.high = 0.95)
#obtain marker genes detected by CAM for estimating A
MGlist <- MGsforA(rCAM, K = 3)

#obtain data and run CAM step by step
rPrep <- CAMPrep(data, dim.rdc= 3, thres.low = 0.30, thres.high = 0.95)
rMGC <- CAMMGCluster(3, rPrep)
#obtain marker genes detected by CAM for estimating A
MGlist <- MGsforA(PrepResult = rPrep, MGResult = rMGC)
```

`MGstatistic` *Statistics for ranking marker genes*

Description

This function computes One-Versus-Everyone Fold Change (OVE-FC) from subpopulation-specific expression profiles. Bootstrapping is optional.

Usage

```
MGstatistic(data, A = NULL, boot.alpha = NULL, nboot = 1000,
  cores = NULL)
```

Arguments

`data`	A data set that will be internally coerced into a matrix. Each row is a gene and each column is a sample. Data should be in non-log linear space with non-negative numerical values (i.e. >= 0). Missing values are not supported. All-zero rows will be removed internally.
`A`	When data are mixture expression profiles, A is estimated proportion matrix or prior proportion matrix. When data are pure expression profiles, A is a phenotype vector to indicate which subpopulation each sample belongs to.
`boot.alpha`	Alpha for bootstrapped OVE-FC confidence interval. The default is 0.05.
`nboot`	The number of boots.
`cores`	The number of system cores for parallel computing. If not provided, the default back-end is used.

Details

This function calculates OVE-FC and bootstrapped OVE-FC which can be used to identify markers from all genes.

Value

A data frame containing the following components:

`idx`	Numbers or phenotypes indicating which subpopulation each gene could be a marker for. If A is a proportion matrix without column name, numbers are returned. Otherwise, phenotypes.
`OVE.FC`	One-versus-Everyone fold change (OVE-FC)
`OVE.FC.alpha`	lower confidence bound of bootstrapped OVE-FC at alpha level.

Examples

```
#data are mixture expression profiles, A is proportion matrix
data(ratMix3)
MGstat <- MGstatistic(ratMix3$X, ratMix3$A)
## Not run:
MGstat <- MGstatistic(ratMix3$X, ratMix3$A, boot.alpha = 0.05) #enable boot

## End(Not run)

#data are pure expression profiles without replicates
MGstat <- MGstatistic(ratMix3$S) #boot is not applicable
## Not run:
#data are pure expression profiles with phenotypes
S <- matrix(rgamma(3000,0.1,0.1), 1000, 3)
S <- S[, c(1,1,1,2,2,2,3,3,3,3)] + rnorm(1000*10, 0, 0.5)
MGstat <- MGstatistic(S, c(1,1,1,2,2,2,3,3,3,3), boot.alpha = 0.05)

## End(Not run)
```

PCAmat *Dimension-reduction loading matrix accessor*

Description

Accessor to Dimension-reduction loading matrix.

Usage

```
PCAmat(x, ...)

## S4 method for signature 'CAMObj'
PCAmat(x)

## S4 method for signature 'CAMPrepObj'
PCAmat(x)
```

Arguments

x	a CAMObj object or a CAMPrepObj object
...	additional argument list.

Value

The matrix whose rows are loading vectors for dimension reduction.

Examples

```
#obtain data
data(ratMix3)
data <- ratMix3$X

rCAM <- CAM(data, K = 3, dim.rdc = 3, thres.low = 0.30, thres.high = 0.95)
W <- PCAmat(rCAM)
W <- PCAmat(slot(rCAM, "PrepResult"))
```

---

ratMix3 *Gene expression data downsampled from GSE19380*

Description

Rat brain, liver and lung biospecimens derived from one animal at the cRNA homogenate level in different proportions. 3 technical replicates each. We downsample the original data to 10000 probes/probesets and 7 mixtures. Proportions used in experiments and pure expression profiles are also included.

Usage

```
data(ratMix3)
```

Format

A list with three matrices: mixture profiles (X), mixing proportions (A) and pure profiles (S).

References

Shen-Orr et al. (2010) Nat Methods 2010 Apr;7(4):287-9. PMID: 20208531

---

redoASest *Re-estimate A, S matrix*

Description

This function re-estimates proportion and expression matrix iteratively by Alternating Least Square (ALS) method. The initial values are from markers or known proportion matrix or known expression matrix,

Usage

```
redoASest(data, MGlist, A = NULL, S = NULL, generalNMF = FALSE,
  maxIter = 2, methy = FALSE)
```

Arguments

data	A data set that will be internally coerced into a matrix. Each row is a gene and each column is a sample. data should be in non-log linear space with non-negative numerical values (i.e. >= 0). Missing values are not supported. All-zero rows will be removed internally.
MGlist	A list of vectors, each of which contains CAM-detected markers and/or prior markers for one subpopulation.
A	Initial proportion matrix. If NULL, it will be estimated from initial expression matrix. If initial expression matrix is also NULL, it will be estimated from MGlist using AfromMarkers.
S	Initial expression matrix. If NULL, it will be estimated from initial proportion matrix.
generalNMF	If TRUE, the decomposed proportion matrix has no sum-to-one constraint for each row. Without this constraint, the scale ambiguity of each column vector in proportion matrix will not be removed. The default is FALSE.
maxIter	maximum number of iterations for Alternating Least Square (ALS) method. The default is 2. If zero, ALS is not applied.
methy	Should be TRUE when dealing with methylation data, whose expression levels are confined between 0 and 1.

Details

If only markers are provided, they are used to estimate initial proportion matrix and then expression matrix. If proportion matrix or expression matrix is provided, it will be treated as initial matrix to estimate the other one. Then Alternating Least Square (ALS) method is applied to estimate two matrix alternatively. Note only markers' squared errors will be counted in ALS, which facilitates (1) faster computational running time and (2) a greater signal-to-noise ratio owing to markers' discriminatory power.

This function can be used to refine CAM estimation or perform supervised deconvolution. Note that allowing too many iterations may bring the risk of a significant deviation from initial values.

Value

A list containing the following components:

Aest	Proportion matrix after re-estimation and possible refinement.
Sest	expression matrix after re-estimation and possible refinement.
mse	Mean squared error (i.e. mean of reconstruction errors) for input markers

Examples

```
#obtain data and run CAM
data(ratMix3)
data <- ratMix3$X
rCAM <- CAM(data, K = 3, dim.rdc= 3, thres.low = 0.30, thres.high = 0.95)
#obtain marker genes detected by CAM for estimating A
MGlist <- MGsforA(rCAM, K = 3)
```

```
#Re-estimation based on marker list
rre <- redoASest(data, MGlist, maxIter = 10)
Aest <- rre$Aest #re-estimated A matrix
Sest <- rre$Sest #re-estimated S matrix

#Re-estimation with initial A matrix
rre <- redoASest(data, MGlist, A=ratMix3$A, maxIter = 10)

#Re-estimation with initial S matrix
rre <- redoASest(data, MGlist, S=ratMix3$S, maxIter = 10)
```

`reselectMG` *Reselect markers by thresholding*

Description

This function generates a new list of markers based on initially detected markers by CAM and/or prior markers.

Usage

```
reselectMG(data, MGlist, fc.thres = "q0.5", err.thres = NULL)
```

Arguments

`data`	A data set that will be internally coerced into a matrix. Each row is a gene and each column is a sample. data should be in non-log linear space with non-negative numerical values (i.e. >= 0). Missing values are not supported. All-zero rows will be removed internally.
`MGlist`	A list of vectors, each of which contains CAM-detected markers and/or prior markers for one subpopulation.
`fc.thres`	The lower threshold of fold change to select markers, an absolute value (e.g. 10) or a quantile value after 'q' (e.g. 'q0.5', the median of fold changes of one subpopulation's input markers). Each subpopulation can have its own threshold if a vector is provided. The default is 'q0.5'. If NULL, still use the input markers.
`err.thres`	The upper threshold of reconstruction error to select markers, an absolute value or a quantile value after 'q' (e.g. 'q0.5', the median of reconstruction errors of one subpopulation's input markers; 'q1.2', the maximum error times 1.2). Each subpopulation can have its own threshold if a vector is provided. The default is NULL, which means no such a threshold is applied.

Details

Considering some meaningful markers may be mistakenly filtered by preprocessing and thus missed by `CAM`, this function use the input marker gene list to estimate proportions by `AfromMarkers` and then estimate expression levels. Next, a new list of markers are generated by fold change threshold and reconstruction error threshold.

The input marker genes could also be from other supervised detection and/or from literatures. This function reselects a list of marker genes based on the input.

Value

A list of vectors, each of which contains new selected markers for one subpopulation.

Examples

```
#obtain data and run CAM
data(ratMix3)
data <- ratMix3$X
rCAM <- CAM(data, K = 3, dim.rdc= 3, thres.low = 0.30, thres.high = 0.95)
#obtain marker genes detected by CAM with a fixed K
MGlist <- MGsforA(rCAM, K = 3)
#Reselect markers from all genes
MGlist.re <- reselectMG(data, MGlist, fc.thres='q0.5')
MGlist.re <- reselectMG(data, MGlist, fc.thres='q0.5', err.thres='q0.95')
```

---

sffsHull *Sequential Forward Floating Selection (SFFS) to approximate convex hull*

Description

This function detects the corners of convex hull by greedy search. It will be used to reduce the number of candidate corners and thus reduce the time complexity in the further exhaustive search by cornerSort.

Usage

```
sffsHull(Xall, Aall, Kmax, deltaK = 8)
```

Arguments

Xall	Data in a matrix. Each column is a data point. The cost is computed based on all data points.
Aall	Candiatie corners in a matrix. Each column is a candidate corner.
Kmax	The target number of corners to be selected.
deltaK	The extra number of corners that need to be searched. The default is 8 and will be truncated based on all the available corners. SFFS runs until a corner set of cardinality (Kmax + deltak) is obtained. The set of cardinality Kmax might be improved during backtracking from extra corners.

Details

The Sequential Floating Forward selection (SFFS) is one of greedy search methods for feature selection. With sum of margin-of-errors as cost function and candidate corners as features, SFFS is used to select best Kmax corners to form an approximated hull. The best subset of candidate corners is initialized as the empty set and at each step a new corner is added. After that, the algorithm searches for corner that can be removed from the best subset until the cost function does not decrease.

Value

A list with length (Kmax + deltak). Each component is a vector of the corner indices of the SFFS-selected subset with certain cardinality.

Examples

```
data <- matrix(c(0.1,0.2,1.0,0.0,0.0,0.5,0.3,
                 0.1,0.7,0.0,1.0,0.0,0.5,0.3,
                 0.8,0.1,0.0,0.0,1.0,0.0,0.4), nrow =3, byrow = TRUE)
rsffs <- sffsHull(data, data, 3)
rsffs <- sffsHull(data, data[,1:5], 3)
```

---

simplexplot *The plot of scatter simplex*

Description

This function shows scatter simplex of mixture expressions.

Usage

```
simplexplot(data, A, MGlist = NULL, data.extra = NULL,
  corner.order = NULL, col = "gray", pch = 1, cex = 0.8,
  mg.col = "red", mg.pch = 1, mg.cex = 1.2, ex.col = "black",
  ex.pch = 19, ex.cex = 1.5, ...)
```

Arguments

data	A data set that will be internally coerced into a matrix. Each row is a gene and each column is a sample. Data should be in non-log linear space with non-negative numerical values (i.e. >= 0). Missing values are not supported. All-zero rows will be removed internally.
A	Prior/Estimated proportion matrix.
MGlist	A list of vectors, each of which contains known markers and/or CAM-detected markers for one subpopulation.
data.extra	Extra data points to be shown in the simplex plot, e.g. the points associated with prior/estimated proportion vectors. The format should be consistent with data, so prior/estimated proportion matrix needs to be transposed before as the input.

corner.order	The order to show simplex corners counterclockwise.
col	The color for data points. The default is "gray".
pch	The symbol/character for data points. The default is 1.
cex	The symbol/character expansion for data points. The default is 0.8.
mg.col	The colors for marker genes. Marker genes of one subpopulation could have their own color if a vector is provided. The default is "red".
mg.pch	The symbols/characters for marker genes. Marker genes of one subpopulation could have their own symbol/character if a vector is provided. The default is 1.
mg.cex	The symbol/character expansion for marker genes. The default is 1.2.
ex.col	The colors for extra data points. Each data point could have its own color if a vector is provided. The default is "black".
ex.pch	The symbols/characters for extra data points. Each data point could have its own symbol/character if a vector is provided. The default is 19.
ex.cex	The symbol/character expansion for extra data points. The default is 1.5.
...	All other arguments are passed to the plotting command.

Details

This function can show the scatter simplex and detected marker genes in a 2D plot. The corners in the high-dimensional simplex will still locate at extreme points of low-dimensional simplex. These corners will follow the order set by `corner.order` to display in the plot counterclockwise.

Value

A plot to the current device.

Examples

```
#obtain data, A matrix, marker genes
data(ratMix3)
data <- ratMix3$X
A <- ratMix3$A
pMGstat <- MGstatistic(ratMix3$S, c("Liver","Brain","Lung"))
pMGlist.FC <- lapply(c("Liver","Brain","Lung"), function(x)
    rownames(pMGstat)[pMGstat$idx == x & pMGstat$OVE.FC > 10])

#plot simplex for data
simplexplot(data, A)
simplexplot(data, A, MGlist = pMGlist.FC) #Color marker genes in the plot
simplexplot(data, A, MGlist = pMGlist.FC,
    data.extra = t(A)) #show the location of proportion vectors in the plot

#set different corner order and colors
simplexplot(data, A, MGlist = pMGlist.FC, corner.order = c(2,1,3),
            col = "blue", mg.col = c("red","orange","green"))
```

XWProj *Perspective projection to obtain simplex*

Description

This function reduces data dimension by loading matrix and then project dimension-reduced data to the hyperplane orthogonal to c(1,0,...,0), i.e., the first axis in the new coordinate system..

Usage

```
XWProj(data, W)
```

Arguments

data	A data set that will be internally coerced into a matrix. Each row is a gene and each column is a sample. Missing values are not supported. All-zero rows will be removed internally.
W	The matrix whose rows are loading vectors; should be obtained from CAM/CAMPrep function with accessor method PCAmat.

Details

This function can project gene expression vectors to simplex plot generated by CAM/CAMPrep. Using slot Xproj in "CAMPrepObj" can only show the simplex of genes after filtering. This function helps observe all genes in simplex plot.

Value

The data after perspective projection.

Examples

```
#obtain data
data(ratMix3)
data <- ratMix3$X

#preprocess data
rPrep <- CAMPrep(data, dim.rdc = 3, thres.low = 0.50, thres.high = 0.90)

#obtain simplex
Xproj <- XWProj(data, PCAmat(rPrep))
#plot simplex in 3d space
#plot3d(Xproj[,-1]) #The first dimension is constant after projection
```

Index