Package ‘M3C’

October 16, 2018

Title Monte Carlo Consensus Clustering

Version 1.2.0

Description Genome-wide data is used to stratify patients into classes using class discovery algorithms. However, we have observed systematic bias present in current state-of-the-art methods. This arises from not considering reference distributions while selecting the number of classes (K). As a solution, we developed a consensus clustering-based algorithm with a hypothesis testing framework called Monte Carlo consensus clustering (M3C). M3C uses a multi-core enabled Monte Carlo simulation to generate null distributions along the range of K which are used to calculate p values to select its value. P values beyond the limits of the simulation are estimated using a beta distribution. M3C can quantify structural relationships between clusters and uses spectral clustering to deal with non-gaussian and imbalanced structures.

Depends R (>= 3.4.0)

License AGPL-3

Encoding UTF-8

LazyData true

Imports ggplot2, Matrix, doSNOW, NMF, RColorBrewer, cluster, parallel, foreach, doParallel, matrixcalc, dendextend, sigclust

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.0.1

biocViews Clustering, GeneExpression, Transcription, RNASeq, Sequencing

git_url https://git.bioconductor.org/packages/M3C

git_branch RELEASE_3_7

git_last_commit 3bfc010

git_last_commit_date 2018-04-30

Date/Publication 2018-10-15

Author Christopher John [aut, cre]

Maintainer Christopher John <chris.r.john86@gmail.com>

R topics documented:

`clustersim` *clustersim: A cluster simulator for testing clustering algorithms*

Description

clustersim: A cluster simulator for testing clustering algorithms

Usage

```
clustersim(n, n2, r, K, alpha, wobble, redp = NULL, print = FALSE,
  seed = NULL)
```

Arguments

`n`	Numerical value: The number of samples, it must be square rootable
`n2`	Numerical value: The number of features
`r`	Numerical value: The radius to define the initial circle (use approx n/100)
`K`	Numerical value: How many clusters to simulate
`alpha`	Numerical value: How far to pull apart the clusters
`wobble`	Numerical value: The degree of noise to add to the sample co ordinates
`redp`	Numerical value: The fraction of samples to remove from one cluster
`print`	Logical flag: whether to print the PCA into current directory
`seed`	Numerical value: fixes the seed if you want to repeat results

Value

A list: containing 1) matrix with simulated data in it

Examples

```
res <- clustersim(225, 900, 8, 4, 0.75, 0.025, redp = NULL, print = TRUE, seed=123)
```

desx *GBM clinical annotation data*

Description

This is the clinical annotation data from the GBM dataset, it contains the class of the tumour which is one of: classical, mesenchymal, neural, proneural. It is a data frame with 2 columns and 50 rows.

Author(s)

Chris John <chris.r.john86@gmail.com>

References

Verhaak, Roel GW, et al. "Integrated genomic analysis identifies clinically relevant subtypes of glioblastoma characterized by abnormalities in PDGFRA, IDH1, EGFR, and NF1." Cancer cell 17.1 (2010): 98-110.

M3C *M3C: Monte Carlo Consensus Clustering*

Description

This function runs M3C, which is a consensus clustering tool with hypothesis testing. The basic idea is to use a multi-core enabled Monte Carlo simulation to drive the creation of a null distribution of stability scores. The monte carlo simulations maintains the correlation structure of the input data. Then the null distribution is used to compare the reference scores with the real scores and a empirical p value is calculated for every value of K. We also use the relative cluster stability index as an alternative metric which is just based on a comparison against the reference mean, the advantage being it requires fewer iterations. Small p values are estimated cheaply using a beta distribution that is inferred using parameter estimates from the Monte Carlo simulation.

Usage

```
M3C(mydata, montecarlo = TRUE, cores = 1, iters = 100, maxK = 10,
  des = NULL, ref_method = c("reverse-pca", "chol"), repsref = 100,
  repsreal = 100, clusteralg = c("pam", "km", "spectral"),
  distance = "euclidean", pacx1 = 0.1, pacx2 = 0.9, printres = FALSE,
  printheatmaps = FALSE, showheatmaps = FALSE, seed = NULL,
  removeplots = FALSE, dend = FALSE)
```

Arguments

mydata	Data frame or matrix: Contains the data, with samples as columns and rows as features
montecarlo	Logical flag: whether to run the Monte Carlo simulation or not (recommended: TRUE)
cores	Numerical value: how many cores to split the monte carlo simulation over

`iters` Numerical value: how many Monte Carlo iterations to perform (default: 100, recommended: 100-1000)

`maxK` Numerical value: the maximum number of clusters to test for, K (default: 10)

`des` Data frame: contains annotation data for the input data for automatic reordering (optional)

`ref_method` Character string: refers to which reference method to use (recommended: leaving as default)

`repsref` Numerical value: how many reps to use for the Monte Carlo reference data (suggest 100)

`repsreal` Numerical value: how many reps to use for the real data (recommended: 100)

`clusteralg` String: dictates which algorithm to use for M3C (recommended: leaving as default)

`distance` String: dictates which distance metric to use for M3C (recommended: leaving as default)

`pacx1` Numerical value: The 1st x co-ordinate for calculating the pac score from the CDF (default: 0.1)

`pacx2` Numerical value: The 2nd x co-ordinate for calculating the pac score from the CDF (default: 0.9)

`printres` Logical flag: whether to print all results into current directory

`printheatmaps` Logical flag: whether to print all the heatmaps into current directory

`showheatmaps` Logical flag: whether to show the heatmaps on screen (can be slow)

`seed` Numerical value: fixes the seed if you want to repeat results, set the seed to 123 for example here

`removeplots` Logical flag: whether to remove all plots (recommended: leaving as default)

`dend` Logical flag: whether to compute the dendrogram and p values for the optimal K or not

Value

A list, containing: 1) the stability results and 2) all the output data (another list) 3) reference stability scores (see vignette for more details on how to easily access)

Examples

```
res <- M3C(mydata, cores=1, iters=100, ref_method = 'reverse-pca', montecarlo = TRUE,printres = FALSE,
maxK = 10, showheatmaps = FALSE, repsreal = 100, repsref = 100,printheatmaps = FALSE, seed = 123, des = desx)
```

`mydata` *GBM expression data*

Description

This is the expression data from the GBM dataset. It is a data frame with 50 columns and 1740 rows.

Author(s)

Chris John <`chris.r.john86@gmail.com`>

References

Verhaak, Roel GW, et al. "Integrated genomic analysis identifies clinically relevant subtypes of glioblastoma characterized by abnormalities in PDGFRA, IDH1, EGFR, and NF1." Cancer cell 17.1 (2010): 98-110.

`pca` *pca: A principal component analysis function*

Description

pca: A principal component analysis function

Usage

```
pca(mydata, K = FALSE, printres = FALSE, labels = FALSE)
```

Arguments

`mydata`	Data frame or matrix or M3C results object: if dataframe/matrix should have samples as columns and rows as features
`K`	Numerical value: if running on the M3C results object, which value was the optimal K?
`printres`	Logical flag: whether to print the PCA into current directory
`labels`	Factor: if we want to just display gender for example

Value

A PCA plot object

Examples

```
PCA <- pca(mydata)
```

Index