Package: ClueR (via r-universe)

September 12, 2024

Type Package

Title Cluster Evaluation

Version 1.4.2

Date 2023-11-14

Author Pengyi Yang

Maintainer Pengyi Yang <yangpy7@gmail.com>

Description CLUster Evaluation (CLUE) is a computational method for identifying optimal number of clusters in a given time-course dataset clustered by cmeans or kmeans algorithms and subsequently identify key kinases or pathways from each cluster. Its implementation in R is called ClueR. See README on <https://github.com/PYangLab/ClueR> for more details. P Yang et al. (2015) <doi:10.1371/journal.pcbi.1004403>.

License GPL-3

Depends R (>= 2.10.0)

Imports e1071, parallel, stats, graphics, grDevices

Encoding UTF-8

RoxygenNote 7.2.3

Repository https://pyanglab.r-universe.dev

RemoteUrl https://github.com/pyanglab/cluer

RemoteRef HEAD

RemoteSha 6dbfcaa079c58e11918e944c8ecd0b3392bbc619

Contents

`ClueR-package` *CLUster Evaluation R package*

Description

CLUster Evaluation (or "CLUE") is an R package for detecting kinases or pathways from a given time-series phosphoproteomics or gene expression dataset clustered by cmeans or kmeans algorithms. It firstly identifies the optimal number of clusters in the time-servies dataset; Then, it partition the dataset based on the optimal number of clusters determined in the first step; It finally detects kinases or pathways enriched in each cluster from optimally partitioned dataset.

The above three steps rely extensively on Fisher's exact test, Fisher's combined statistics, cluster regularisations, and they are performed against a user-specified reference annotation database such phosphoSitePlus in the case of phosphoproteomics data or KEGG in the case of gene expression data. There is a selection of built-in annotation databases for both phosphoproteomics data and gene expression data but users can supply their own annotation database.

CLUE was initially designed for analysing time-course phosphoproteomics dataset using kinase-substrate annotation as reference (e.g. PhosphoSitePlus). It is now extended to identify key pathways from time-series microarray, RNA-seq or proteomics datasets by searching and testing against gene set annotation databases such as KEGG, GO, or Reactome etc.

Previously published phosphoproteomics dataset and gene expression dataset are included in the package to demonstrate how to use CLUE package.

See help from the main function by typing '?runClue' for more details and examples on how to use CLUE.

You can also install the latest development version from github with:

devtools::install_github("PengyiYang/ClueR")

Make sure that you have Rtools install in your system for building the package from the source.

Details

Package:	CLUE
Type:	Package
Version:	1.4.2
Date:	2023-11-14
License:	GPL-3

Author(s)

Pengyi Yang

References

Yang P, Zheng X, Jayaswal V, Hu G, Yang JYH, Jothi R (2015) Knowledge-Based Analysis for Detecting Key Signaling Events from Time-Series Phosphoproteomics Data. PLoS Comput Biol 11(8): e1004403.

---

adipocyte *Mouse adipocyte differentiation gene expression (microarray) data*

Description

The data object contains a time-course gene expression profiling of the mouse adipocytes through differentiation. For details please refer to the article:

Ma X, Yang P, Kaplan WH, Lee BH, Wu LE, Yang JY, Yasunaga M, Sato K, Chisholm DJ, James DE. ISL1 regulates peroxisome proliferator-activated receptor gamma activation and early adipogenesis via bone morphogenetic protein 4-dependent and-independent mechanisms. Molecular and Cellular Biology. 2014 Oct 1;34(19):3607-17.

---

clustEnrichment *Cluster enrichment test*

Description

Takes a clustering object generted by cmeans or kmeans algorithm and determine the enrichment of each cluster and then the overall enrichment of this clustering object based on an annotation file.

Usage

```
clustEnrichment(
  clustObj,
  annotation,
  effectiveSize,
  pvalueCutoff = 0.05,
  universe = NULL
)
```

Arguments

clustObj	the clustering object generated by cmeans or kmeans.
annotation	a list with names correspond to kinases and elements correspond to substrates belonging to each kinase.
effectiveSize	the size of kinase-substrate groups to be considered for calculating enrichment. Groups that are too small or too large will be removed from calculating overall enrichment of the clustering.
pvalueCutoff	a pvalue cutoff for determining which kinase-substrate groups to be included in calculating overall enrichment of the clustering.
universe	the universe of genes/proteins/phosphosites etc. that the enrichment is calculated against.

Value

a list that contains both the p-value indicating the overall enrichment and a sublist that details the enrichment of each individual cluster.

Examples

```
# simulate a time-series data with six distinctive profile groups and each group with
# a size of 500 phosphorylation sites.
simuData <- temporalSimu(seed=1, groupSize=500, sdd=1, numGroups=4)

# create an artificial annotation database. Generate 100 kinase-substrate groups each
# comprising 50 substrates assigned to a kinase.
# among them, create 5 groups each contains phosphorylation sites defined to have the
# same temporal profile.
kinaseAnno <- list()
groupSize <- 500
for (i in 1:5) {
 kinaseAnno[[i]] <- paste("p", (groupSize*(i-1)+1):(groupSize*(i-1)+50), sep="_")
}

for (i in 6:100) {
 set.seed(i)
 kinaseAnno[[i]] <- paste("p", sample.int(nrow(simuData), size = 50), sep="_")
}
names(kinaseAnno) <- paste("KS", 1:100, sep="_")
```

```
# testing enrichment of clustering results by partition the data into six clusters
# using cmeans algorithm.
clustObj <- e1071::cmeans(simuData, centers=6, iter.max=50, m=1.25)
clustEnrichment(clustObj, annotation=kinaseAnno, effectiveSize=c(5, 100), pvalueCutoff=0.05)
```

`clustOptimal` *Generate optimal clustering*

Description

Takes a clue output and generate the optimal clustering of the time-course data.

Usage

```
clustOptimal(
  clueObj,
  rep = 5,
  user.maxK = NULL,
  effectiveSize = NULL,
  pvalueCutoff = 0.05,
  visualize = TRUE,
  universe = NULL,
  mfrow = c(1, 1)
)
```

Arguments

`clueObj`	the output from runClue.
`rep`	number of times (default is 5) the clustering is to be repeated to find the best clustering result.
`user.maxK`	user defined optimal k value for generating optimal clustering. If not provided, the optimal k that is identified by clue will be used.
`effectiveSize`	the size of kinase-substrate groups to be considered for calculating enrichment. Groups that are too small or too large will be removed from calculating overall enrichment of the clustering.
`pvalueCutoff`	a pvalue cutoff for determining which kinase-substrate groups to be included in calculating overall enrichment of the clustering.
`visualize`	a boolean parameter indicating whether to visualize the clustering results.
`universe`	the universe of genes/proteins/phosphosites etc. that the enrichment is calculated against. The default are the row names of the dataset.
`mfrow`	control the subplots in graphic window.

Value

return a list containing optimal clustering object and enriched kinases or gene sets.

Examples

```
# simulate a time-series data with 4 distinctive profile groups and each group with
# a size of 50 phosphorylation sites.
simuData <- temporalSimu(seed=1, groupSize=50, sdd=1, numGroups=4)

# create an artificial annotation database. Generate 20 kinase-substrate groups each
# comprising 10 substrates assigned to a kinase.
# among them, create 4 groups each contains phosphorylation sites defined to have the
# same temporal profile.
kinaseAnno <- list()
groupSize <- 50
for (i in 1:4) {
 kinaseAnno[[i]] <- paste("p", (groupSize*(i-1)+1):(groupSize*(i-1)+10), sep="_")
}

for (i in 5:20) {
 set.seed(i)
 kinaseAnno[[i]] <- paste("p", sample.int(nrow(simuData), size = 10), sep="_")
}
names(kinaseAnno) <- paste("KS", 1:20, sep="_")

# run CLUE with a repeat of 2 times and a range from 2 to 7
set.seed(1)
clueObj <- runClue(Tc=simuData, annotation=kinaseAnno, rep=5, kRange=2:7)

# visualize the evaluation outcome
xl <- "Number of clusters"
yl <- "Enrichment score"
boxplot(clueObj$evlMat, col=rainbow(ncol(clueObj$evlMat)), las=2, xlab=xl, ylab=yl, main="CLUE")
abline(v=(clueObj$maxK-1), col=rgb(1,0,0,.3))

# generate optimal clustering results using the optimal k determined by CLUE
best <- clustOptimal(clueObj, rep=3, mfrow=c(2, 3))

# list enriched clusters
best$enrichList

# obtain the optimal clustering object

  best$clustObj
```

`enrichmentTest` *Fisher's exact test-based enrichment test*

Description

Takes a vector of names representing phosphorylation sites that are partitioned in the same cluster and an kinase-substrate annotation. Test for enrichment of the kinase based on the name vector.

Usage

```
enrichmentTest(clust, annotation, universe, alter = "greater")
```

Arguments

clust	a vector of names representing phosphorylation sites that are partitioned in the same cluster
annotation	a list with names correspond to kinases and elements correspond to substrates belong to each kinase
universe	the universe of names to compare against
alter	indicates the alternative hypothesis and must be one of "two.sided", "greater" or "less"

Value

a matrix that contains enrichment of each kinase based on the input name vector.

---

fuzzPlot *Visualize fuzzy clustering results*

Description

Takes in a time-course matrix and its clustering results as a cmeans clustering object. Produce a plot to visualize the clustering results.

Usage

```
fuzzPlot(
  Tc,
  clustObj,
  mfrow = c(1, 1),
  cols,
  min.mem = 0,
  new.window = FALSE,
  llwd = 3
)
```

Arguments

Tc	a numeric matrix to be clustered. The columns correspond to the time-course and the rows correspond to phosphorylation sites.
clustObj	the clustering of Tc generated from cmeans or kmeans clustering.
mfrow	control the subplots in graphic window.
cols	color palette to be used for plotting. If the color argument remains empty, the default palette is used.

min.mem	phosphorylation sites with membership values below min.mem will not be displayed.
new.window	should a new window be opened for graphics.
llwd	line width. Default is 3.

Examples

```
# load the human ES phosphoprotoemics data (Rigbolt et al. Sci Signal. 4(164):rs3, 2011)
data(hES)
# apply cmeans clustering to partition the data into 11 clusters
clustObj <- e1071::cmeans(hES, centers=11, iter.max=50, m=1.25)
# visualize clustering reuslts
fuzzPlot(hES, clustObj, mfrow = c(3,4))
```

---

hES *Human embryonic stem cell phosphoproteomics data*

Description

The data object contains a time-course phosphoprotoemics profiling of the human embryonic stem cells through differentiation. The differentiation is induced by using PMA and the time points of 30m, 1hr, 6hrs, and 24hrs are measured and the ratio are taken against 0m. For details please refer to the article: Rigbolt et al. Sci Signal. 4(164):rs3, 2011

Pathways.biocarta *Biocarta pathway annotations*

Description

The data object contains the annotations from biocarta pathway database.

Pathways.DAVID *DAVID pathway annotations*

Description

The data object contains the annotations from DAVID database which contains GOBP (GO biological processes) pathways.

Pathways.KEGG *KEGG pathway annotations*

Description

The data object contains the annotations of KEGG pathways.

Pathways.reactome *Reactome pathway annotations*

Description

The data object contains the annotations of reactome pathways.

PhosphoELM.human *Phospho.ELM annotations for human*

Description

The data object contains the annotations of kinases and their conrresponding substrates as phosphorylation sites in human. It is extracted from the Phospho.ELM database.

PhosphoELM.mouse *Phospho.ELM annotations for mouse*

Description

The data object contains the annotations of kinases and their conrresponding substrates as phosphorylation sites in mouse. It is extracted from the Phospho.ELM database.

PhosphoSite.human *PhosphoSitePlus annotations for human*

Description

The data object contains the annotations of kinases and their conrresponding substrates as phosphorylation sites in human. It is extracted from the PhosphoSitePlus database. For details of PhosphoSitePlus, please refer to the article: Hornbeck et al. Nucleic Acids Res. 40:D261-70, 2012

PhosphoSite.mouse *PhosphoSitePlus annotations for mouse*

Description

The data object contains the annotations of kinases and their conrresponding substrates as phosphorylation sites in mouse. It is extracted from the PhosphoSitePlus database. For details of PhosphoSitePlus, please refer to the article: Hornbeck et al. Nucleic Acids Res. 40:D261-70, 2012

runClue *Run CLUster Evaluation*

Description

Takes in a time-course matrix and test for enrichment of the clustering using cmeans or kmeans clustering algorithm with a reference annotation.

Usage

```
runClue(
  Tc,
  annotation,
  rep = 5,
  kRange = 2:10,
  clustAlg = "cmeans",
  effectiveSize = c(5, 100),
  pvalueCutoff = 0.05,
  alpha = 0.5,
  standardise = TRUE,
  universe = NULL
)
```

Arguments

Tc	a numeric matrix to be clustered. The columns correspond to the time-course and the rows correspond to phosphorylation sites.
annotation	a list with names correspond to kinases and elements correspond to substrates belong to each kinase.
rep	number of times the clustering is to be applied. This is to account for variability in the clustering algorithm. Default is 5.
kRange	the range of k to be tested for clustering. Default is 2:10
clustAlg	the clustering algorithm to be used. The default is cmeans clustering.

effectiveSize	the size of annotation groups to be considered for calculating enrichment. Groups that are too small or too large will be removed from calculating overall enrichment of the clustering.
pvalueCutoff	a pvalue cutoff for determining which kinase-substrate groups to be included in calculating overall enrichment of the clustering.
alpha	a regularisation factor for penalizing large number of clusters.
standardise	whether to z-score standardise the input matrix.
universe	the universe of genes/proteins/phosphosites etc. that the enrichment is calculated against. The default are the row names of the dataset.

Value

a clue output that contains the input parameters used for evaluation and the evaluation results. Use ls(x) to see details of output. 'x' be the output here.

Examples

```
## Example 1. Running CLUE with a simulated phosphoproteomics data

## simulate a time-series phosphoproteomics data with 4 clusters and
## each cluster with a size of 100 phosphosites
simuData <- temporalSimu(seed=1, groupSize=100, sdd=1, numGroups=4)

## create an artificial annotation database. Specifically, Generate 50
## kinase-substrate groups each comprising 20 substrates assigned to a kinase.
## Among them, create 5 groups each contains phosphosites defined
## to have the same temporal profile.

kinaseAnno <- list()
groupSize <- 100
for (i in 1:5) {
  kinaseAnno[[i]] <- paste("p", (groupSize*(i-1)+1):(groupSize*(i-1)+20), sep="_")
}

for (i in 6:50) {
  set.seed(i)
  kinaseAnno[[i]] <- paste("p", sample.int(nrow(simuData), size = 20), sep="_")
}
names(kinaseAnno) <- paste("KS", 1:50, sep="_")

## run CLUE with a repeat of 3 times and a range from 2 to 8
set.seed(1)
cl <- runClue(Tc=simuData, annotation=kinaseAnno, rep=3, kRange=2:8,
              standardise = TRUE, universe = NULL)

## visualize the evaluation outcome
boxplot(cl$evlMat, col=rainbow(8), las=2, xlab="# cluster", ylab="Enrichment", main="CLUE")

## generate optimal clustering results using the optimal k determined by CLUE
best <- clustOptimal(cl, rep=3, mfrow=c(2, 3))
```

```
## list enriched clusters
best$enrichList

## obtain the optimal clustering object
best$clustObj

## Example 2. Running CLUE with a phosphoproteomics dataset, discover optimal number of clusters,
## clustering data accordingly, and identify key kinases involved in each cluster.

## load the human ES phosphoprotoemics data (Rigbolt et al. Sci Signal. 4(164):rs3, 2011)
data(hES)
# load the PhosphoSitePlus annotations (Hornbeck et al. Nucleic Acids Res. 40:D261-70, 2012)
# note that one can instead use PhosphoELM database by typing "data(PhosphoELM)".
data(PhosphoSite)

## run CLUE with a repeat of 5 times and a range from 2 to 15
set.seed(1)
cl <- runClue(Tc=hES, annotation=PhosphoSite.human, rep=5, kRange=2:15,
              standardise = TRUE, universe = NULL)

boxplot(cl$evlMat, col=rainbow(15), las=2, xlab="# cluster", ylab="Enrichment", main="CLUE")

best <- clustOptimal(cl, rep=3, mfrow=c(4, 4))

best$enrichList

## Example 3. Running CLUE with a gene expression dataset, discover optimal number of clusters,
## clustering data accordingly, and identify key pathway involved in each cluster.

## load mouse adipocyte gene expression data
# (Ma et al. Molecular and Cellular Biology. 2014, 34(19):3607-17)
data(adipocyte)

## load the KEGG annotations
## note that one can instead use reactome, GOBP, biocarta database
data(Pathways)

## select genes that are differentially expressed during adipocyte differentiation
adipocyte.selected <- adipocyte[adipocyte[,"DE"] == 1,]

## run CLUE with a repeat of 5 times and a range from 10 to 22

set.seed(3)
cl <- runClue(Tc=adipocyte.selected, annotation=Pathways.KEGG, rep=3, kRange=10:20,
              standardise = TRUE, universe = NULL)

xl <- "Number of clusters"
yl <- "Enrichment score"
boxplot(cl$evlMat, col=rainbow(ncol(cl$evlMat)), las=2, xlab=xl, ylab=yl, main="CLUE")
```

`temporalSimu` *Temporal data simulation*

Description

This function simulates time-series data using 14 pre-defined temporal profile templates. Type 'temporalSimu' to see the details of the templates.

Usage

```
temporalSimu(seed = unclass(Sys.time()), groupSize, sdd, numGroups)
```

Arguments

`seed`	to seed the simulation. Default is current system time.
`groupSize`	the number of the temporal profiles to simulate from each template. The total number of profiles will be the number of templates used times the size of each group.
`sdd`	the standard deviation to be used to generate randomness for each temporal profile.
`numGroups`	number of templates to be used for generating data.

Value

a matrix containing simulated time-series dataset.

Examples

```
# simulate a time-series data with four distinctive profile groups and each group with
# a size of 500 phosphorylation sites

simulated.temporal <- temporalSimu(seed=1, groupSize=500, sdd=1, numGroups=4)
```

Index