Package 'CAMO'

February 17, 2018

Type Package

Title Perform congruent analysis among model organisms

Version 1.0

Date 2018-02-13

Author Tianzhou Ma

Maintainer Tianzhou Ma <tianzhou.ma0105@gmail.com>

Description The package performs differential transcriptomic quantification among model organisms both globally and at pathway level, and provide several visualization outputs for more biological insight

License GPL (>= 2)

Depends Rcpp (>= 0.12.9), RcppArmadillo, MASS, snowfall, limma, edgeR, ggrepel, ggplot2, gplots, ConsensusClusterPlus, pathview, KEGG.db, org.Hs.eg.db

Imports Rcpp (>= 0.12.9), RcppArmadillo, MASS, snowfall, limma, edgeR, ggrepel, ggplot2, gplots, ConsensusClusterPlus, pathview, KEGG.db, org.Hs.eg.db

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.0.1

R topics documented:

`bayes` *Run Bayesian analysis for individual pdata*

Description

Run Bayesian analysis for individual pData The `bayes` is function to run Bayesian analysis for individual pdata

Usage

```
bayes(pData, seed = 12345)
```

Arguments

`pData:`	individual pData. The data matrix has to consist of two columns, first being the p-value, second being the log fold change.
`seed:`	seed number.

Value

an MCMC output matrix of signed DE indicator (input for resemblance analysis)

Examples

```
## Not run:
data(hb)
summaryDE <- indDE(data=data,group=group,data.type="microarray",
                  case.label="2", ctrl.label="1")
pData <- summaryDE[,c(3,1)]
MCMCout <- bayes(pData, seed=12345)

## End(Not run)
```

clustDiag *Perform consensus clustering diagnostics to determine optimal K for pathway clustering.*

Description

Perform consensus clustering diagnostics to determine optimal K for pathway clustering The `clustDiag` is function to perform consensus clustering diagnostics for pathway clustering. It will generate consensus CDF and delta area plot and saved to the folder named "clustDiag". These plots will help you determine the optimal number of clusters K for pathway clustering. Run this first before `multiOutput` when "clustPathway" output is chosen.

Usage

```
clustDiag(ARS_pathway)
```

Arguments

`ARS_pathway:`	a list of two data frames: pathway specific ARS values and their permuted p-value (pathway on rows, column being ARS value or the p-values).

Value

stored output in the folder named "clustDiag".

Examples

```
## Not run:
#ARS_pathway from the multiARS step
results <- clustDiag(ARS_pathway)

## End(Not run)
```

hashtb *Hash table*

Description

Hash table and complete pathway names for text mining.

Usage

```
data(hashtb)
```

Examples

```
data(hashtb)
```

hb *Human burn dataset*

Description

Human burn dataset: including both microarray expression data matrix and group label.

Usage

```
data(hb)
```

Examples

```
data(hb)
```

hm_orth *Human mouse ortholog file*

Description

Human mouse ortholog file for merging datasets from the two species.

Usage

```
data(hm_orth)
```

Examples

```
data(hm_orth)
```

hs *Human sepsis dataset*

Description

Human sepsis dataset: including both microarray expression data matrix and group label.

Usage

```
data(hs)
```

Examples

```
data(hs)
```

ht *Human trauma dataset*

Description

Human traum dataset: including both microarray expression data matrix and group label.

Usage

```
data(ht)
```

Examples

```
data(ht)
```

indDE *Differential expression analysis for individual data.*

Description

Differential expression analysis for individual data The `indDE` is function to perform differential expression analysis for individual data

Usage

```
indDE(data, group, data.type, case.label, ctrl.label)
```

Arguments

`data:`	the raw expression data.
`group:`	the group label.
`data.type:`	either "microarray" or "RNAseq"
`case.label:`	label for the case group.
`control.label:`	label for the control group.

Value

summary consisting of log fold change, lfc standard error, p-value, q-value.

Examples

```
## Not run:
data(hb)
summaryDE <- indDE(data=data,group=group,data.type="microarray",
                  case.label="2", ctrl.label="1")

## End(Not run)
```

mb *Mouse burn dataset*

Description

Mouse burn dataset: including both microarray expression data matrix and group label.

Usage

```
data(mb)
```

Examples

```
data(mb)
```

mdsPlotRShiny *Analysis results for multiple pairs: visualization outputs*

Description

Plot mds plot (function only for Rshiny)

Usage

```
mdsPlotRShiny(mcmc.merge.list, dataset.names, select.pathway.list, ARS_pathway,
  hashtb = NULL, pathways = NULL, optK = NULL)
```

Arguments

mcmc.merge.list:	a list of merged MCMC output matrices.
dataset.names:	a vector of dataset names.
select.pathway.list:	a list of selected pathways (containing gene components).
ARS_pathway:	a list of two data frames: pathway specific ARS values and their permuted p-value (pathway on rows, column being ARS value or the p-values).
output:	five options: "clustPathway" (pathway clustering),"mdsModel"(model MDS plot),"clustModel" (model clustering output), "genePM" (generating heatmap of gene posterior mean),"keggView" (generating kegg pathway topology, human KEGG only). For details, please refer to manuscript. cannot be empty.
hashtb:	hash table for text mining.
pathways:	complete pathway names for text mining.
keggViewSelect:	which two datasets to view in KEGG topology.
optK:	Optimal number of clusters based on clustering diagnostic results. For "clustPathway" output only.
kegg_pathname:	KEGG pathway name list. For "keggView" only.
hs_gene_id:	Human sapiens gene id. For "keggView" only.

Value

stored output in created folders.

Examples

```
## Not run:
#mcmc.merge.list from the merge step
#select.pathway.list from the pathSelect step
#ARS_pathway from the multiARS step
data(hashtb) #include hashtb & pathways
dataset.names <- c("hb","hs","ht","mb","ms","mt")
library(KEGG.db)
kegg_pathname <- unlist(as.list(KEGGPATHID2NAME))
library("org.Hs.eg.db")
hs_gene_id <- unlist(mget(x=rownames(mcmc.merge.list[[1]]),
envir=org.Hs.egALIAS2EG))
multiOutput(mcmc.merge.list,dataset.names,select.pathway.list,
ARS_pathway, output=c("clustPathway","mdsModel","clustModel","genePM","keggView"),
hashtb=hashtb,pathways=pathways,keggViewSelect = c(1,4),optK=7)

## End(Not run)
```

`mergeMCMC` *Merge multiple MCMCout datasets preparing for cross-species analysis*

Description

Merge multiple MCMCout datasets The `mergeMCMC` is function to merge multiple MCMCout datasets by matching orthologs

Usage

```
mergeMCMC(mcmc.list, species, ortholog.db, ortholog.file = NULL,
  reference = 1)
```

Arguments

`mcmc.list:`	a list of MCMC output matrices.
`species:`	a vector specie names of same length as mcmc.list.
`ortholog.db:`	the ortholog object (in R environment)
`ortholog.file:`	the ortholog file to be imported
`reference:`	the index of the reference data, the outputted merged list will be named using the rownames of this data.

Value

an merged list of multiple MCMCout datasets (with same number of rows and rownames)

Examples

```
## Not run:
data(hb)
summaryDE <- indDE(data=data,group=as.factor(group),data.type="microarray",
                  case.label="2", ctrl.label="1")
hb_pData <- summaryDE[,c(3,1)]
hb_MCMCout <- bayes(hb_pData, seed=12345)
data(hs)
summaryDE <- indDE(data=data,group=as.factor(group),data.type="microarray",
                  case.label="2", ctrl.label="1")
hs_pData <- summaryDE[,c(3,1)]
hs_MCMCout <- bayes(hs_pData, seed=12345)
data(ht)
summaryDE <- indDE(data=data,group=as.factor(group),data.type="microarray",
                  case.label="2", ctrl.label="1")
ht_pData <- summaryDE[,c(3,1)]
ht_MCMCout <- bayes(ht_pData, seed=12345)
data(mb)
summaryDE <- indDE(data=data,group=as.factor(group),data.type="microarray",
                  case.label="2", ctrl.label="1")
mb_pData <- summaryDE[,c(3,1)]
mb_MCMCout <- bayes(mb_pData, seed=12345)
data(ms)
summaryDE <- indDE(data=data,group=as.factor(group),data.type="microarray",
                  case.label="2", ctrl.label="1")
ms_pData <- summaryDE[,c(3,1)]
ms_MCMCout <- bayes(ms_pData, seed=12345)
data(mt)
summaryDE <- indDE(data=data,group=as.factor(group),data.type="microarray",
                  case.label="2", ctrl.label="1")
mt_pData <- summaryDE[,c(3,1)]
mt_MCMCout <- bayes(mt_pData, seed=12345)

#1. single pair example
mcmc.list <- list(hb_MCMCout,mb_MCMCout)
species <- c("human","mouse")
data(hm_orth)
mcmc.merge.list <- mergeMCMC(mcmc.list,species = species,
ortholog.db = hm_orth, reference=1)

#2. multiple pairs example
mcmc.list <- list(hb_MCMCout,hs_MCMCout,ht_MCMCout,
                  mb_MCMCout,ms_MCMCout,mt_MCMCout)
species <- c(rep("human",3), rep("mouse",3))
data(hm_orth)
mcmc.merge.list <- mergeMCMC(mcmc.list,species = species,
ortholog.db = hm_orth, reference=1)

## End(Not run)
```

---

mergeRaw *Merge multiple raw datasets preparing for cross-species analysis*

Description

Merge multiple raw datasets The mergeRaw is function to merge multiple raw datasets by matching orthologs

Usage

```
mergeRaw(data.list, species, ortholog.db, ortholog.file = NULL,
  reference = 1, unique = T)
```

Arguments

data.list:	a list of data matrices.
species:	a vector specie names of same length as data.list.
ortholog.db:	the ortholog object (in R environment)
ortholog.file:	the ortholog file to be imported
reference:	the index of the reference data, the outputted merged list will be named using the rownames of this data.
unique:	logical value indicating whether to take only the "one to one" unique match (T) or allow "multiple to one" and "one to multiple" match (F)

Value

an merged list of rdatasets (with same number of rows and rownames)

Examples

```
## Not run:
data(hb)
hb.data <- data
hb.group <- group
data(hs)
hs.data <- data
hs.group <- group
data(ht)
ht.data <- data
ht.group <- group
data(mb)
mb.data <- data
mb.group <- group
data(ms)
ms.data <- data
ms.group <- group
data(mt)
mt.data <- data
mt.group <- group
data.list <- list(hb.data,hs.data,ht.data,mb.data,ms.data,mt.data)
species <- c(rep("human",3), rep("mouse",3))
data(hm_orth)
data.merge.list <- mergeRaw(data.list,species=species,
ortholog.db = hm_orth, reference=1,unique=T)

## End(Not run)
```

ms *Mouse sepsis dataset*

Description

Mouse sepsis dataset: including both microarray expression data matrix and group label.

Usage

```
data(ms)
```

Examples

```
data(ms)
```

mt *Mouse trauma dataset*

Description

Mouse trauma dataset: including both microarray expression data matrix and group label.

Usage

```
data(mt)
```

Examples

```
data(mt)
```

multiARS_global *Resemblance analysis for multiple pairs: global ARS and its permuted p-value.*

Description

Resemblance analysis for multiple pairs: global ARS and its permuted p-value The `multiARS_global` is function to perform resemblance analysis for multiple pairs, generating global ARS and its permuted p-value.

Usage

```
multiARS_global(mcmc.merge.list, dataset.names, measure = "Fmeasure",
  parallel = F, cpu = 2, B = 50)
```

Arguments

mcmc.merge.list:	a list of merged MCMC output matrices.
dataset.names:	a vector of dataset names.
measure:	three types of ARS measures to be used: "youden", "Fmeasure","geo.mean". Default is "Fmeasure".
parallel:	whether to perform parallel computing in permutation.
cpu:	if parallel=T, how many cpus to be used.
B:	number of permutations.

Value

two lists: global ARS values and its permuted p-value, in addition, the two data matrices are written to the folder named "arsGlobal".

Examples

```
## Not run:
#mcmc.merge.list from the merge step (example 1)
dataset.names <- c("hb","hs","ht","mb","ms","mt")
ARS_global <- multiARS_global(mcmc.merge.list,dataset.names,B=100)

## End(Not run)
```

---

multiARS_pathway *Resemblance analysis for multiple pairs: pathway specific ARS and their permuted p-value.*

Description

Resemblance analysis for multiple pairs: pathway specific ARS and their permuted p-value The multiARS_pathway is function to perform resemblance analysis for multiple pairs, generating pathway specific ARS and their permuted p-value.

Usage

```
multiARS_pathway(mcmc.merge.list, dataset.names, select.pathway.list,
  measure = "Fmeasure", parallel = F, cpu = 2, B = 50)
```

Arguments

mcmc.merge.list:	a list of merged MCMC output matrices.
dataset.names:	a vector of dataset names.
select.pathway.list:	a list of selected pathways (containing gene components).
measure:	three types of ARS measures to be used: "youden", "Fmeasure","geo.mean". Default is "Fmeasure".
parallel:	whether to perform parallel computing in permutation.
cpu:	if parallel=T, how many cpus to be used.
B:	number of permutations.

Value

a list of two data frames: pathway specific ARS values and their permuted p-value (pathway on rows, column being ARS value or the p-values), in addition, both are written to the folder named "arsPathway".

Examples

```
## Not run:
#mcmc.merge.list from the merge step (example 2)
#select.pathway.list from the pathSelect step
dataset.names <- c("hb","hs","ht","mb","ms","mt")
ARS_pathway <- multiARS_pathway(mcmc.merge.list,dataset.names,
select.pathway.list,B=100)

## End(Not run)
```

`multiOutput` *Analysis results for multiple pairs: visualization outputs*

Description

Analysis results for multiple pairs: visualization outputs including overall pathway clustering and output for each pathway The `multiOutput` is function to generate visualization outputs for multiple pairs: including overall pathway clustering outputs, model MDS plot, model clustering output, heatmap of gene posterior mean, kegg pathway topology for each pathway

Usage

```
multiOutput(mcmc.merge.list, dataset.names, select.pathway.list, ARS_pathway,
  output = c("clustPathway", "mdsModel", "clustModel", "genePM", "keggView"),
  hashtb = NULL, pathways = NULL, keggViewSelect = c(1, 2), optK = NULL,
  kegg_pathname = NULL, hs_gene_id = NULL)
```

Arguments

`mcmc.merge.list:`	a list of merged MCMC output matrices.
`dataset.names:`	a vector of dataset names.
`select.pathway.list:`	a list of selected pathways (containing gene components).
`ARS_pathway:`	a list of two data frames: pathway specific ARS values and their permuted p-value (pathway on rows, column being ARS value or the p-values).
`output:`	five options: "clustPathway" (pathway clustering),"mdsModel"(model MDS plot),"clustModel" (model clustering output), "genePM" (generating heatmap of gene posterior mean),"keggView" (generating kegg pathway topology, human KEGG only). For details, please refer to manuscript. cannot be empty.
`hashtb:`	hash table for text mining.
`pathways:`	complete pathway names for text mining.
`keggViewSelect:`	which two datasets to view in KEGG topology.

optK: Optimal number of clusters based on clustering diagnostic results. For "clustPathway" output only.

kegg_pathname: KEGG pathway name list. For "keggView" only.

hs_gene_id: Human sapiens gene id. For "keggView" only.

Value

stored output in created folders.

Examples

```
## Not run:
#mcmc.merge.list from the merge step
#select.pathway.list from the pathSelect step
#ARS_pathway from the multiARS step
data(hashtb) #include hashtb & pathways
dataset.names <- c("hb","hs","ht","mb","ms","mt")
library(KEGG.db)
kegg_pathname <- unlist(as.list(KEGGPATHID2NAME))
library("org.Hs.eg.db")
hs_gene_id <- unlist(mget(x=rownames(mcmc.merge.list[[1]]),
envir=org.Hs.egALIAS2EG))
multiOutput(mcmc.merge.list,dataset.names,select.pathway.list,
ARS_pathway, output=c("clustPathway","mdsModel","clustModel","genePM","keggView"),
hashtb=hashtb,pathways=pathways,keggViewSelect = c(1,4),optK=7)

## End(Not run)
```

---

pathSelect *Select pathways of interest for pathway-level resemblance analysis*

Description

Select pathways of interest The `pathSelect` is function to select pathways of interest for pathway-level resemblance analysis

Usage

```
pathSelect(mcmc.merge.list, pathway.list, pathwaysize.lower.cut = 10,
  pathwaysize.upper.cut = 200, overlapsize.cut = 10, med.de.cut = 5,
  qfisher.cut = 0.01)
```

Arguments

mcmc.merge.list: a list of merged MCMC output matrices.

pathway.list: list of pathway database.

pathwaysize.lower.cut: pathway size lower bound cutoff;

pathwaysize.upper.cut: pathway size upper bound cutoff;

`overlapsize.cut:` lower bound cutoff of overlap size between genes from input data and genes from a pathway.

`med.de.cut:` lower bound cutoff of median number of DE genes in a pathway.

`qfisher.cut:` fisher q-value cutoff from the meta enrichment analysis.

Value

an vector of selected pathway names.

Examples

```
## Not run:
#mcmc.merge.list from the merge step
data(pathwayDB) ## include pathway.list
select.pathway <- pathSelect(mcmc.merge.list,pathway.list)
select.pathway.list <- pathway.list[select.pathway]

## End(Not run)
```

---

pathwayDB *Pathway list*

Description

Pathway list collected from seven major pathway databases for human: GOBP,GOCC,GOMF,Reactome,Biocarta,KEGG, Immunologic.

Usage

```
data(pathwayDB)
```

Examples

```
data(pathwayDB)
```

---

read.groupData *Read group label data.*

Description

Read group label data The `read.groupData` is function to read group label data

Usage

```
read.groupData(group.file, sep = ",", quote = "\"", header = T)
```

Arguments

group.file:	the file name of the group data to be read.
sep:	the field separator character. Values on each line of the file are separated by this character. If sep = "", the separator is 'white space', that is one or more spaces, tabs, newlines or carriage returns. The default is ,. See read.csv.
quote:	the set of quoting characters. To disable quoting altogether, use quote = "". See scan for the behaviour on quotes embedded in quotes. Quoting is only considered for columns read as character, which is all of them unless colClasses is specified. The default is \". See read.csv
header:	a logical value indicating whether the file contains the names of the variables as its first line. If missing, the value is determined from the file format: header is set to TRUE if and only if the first row contains one fewer field than the number of columns. The default is TRUE. See read.csv.

Value

the group label vector.

Examples

```
## Not run:
data(hb)
write.csv(group,file="group.csv")
group <- read.groupData(group.file = "group.csv")

## End(Not run)
```

---

read.pData *Read p-value data.*

Description

Read p-value data The read.pData is function to read p-value data

Usage

```
read.pData(p.file, sep = ",", quote = "\"", header = T)
```

Arguments

p.file:	the file name of the group data to be read.
sep:	the field separator character. Values on each line of the file are separated by this character. If sep = "", the separator is 'white space', that is one or more spaces, tabs, newlines or carriage returns. The default is ,. See read.csv.
quote:	the set of quoting characters. To disable quoting altogether, use quote = "". See scan for the behaviour on quotes embedded in quotes. Quoting is only considered for columns read as character, which is all of them unless colClasses is specified. The default is \". See read.csv
header:	a logical value indicating whether the file contains the names of the variables as its first line. If missing, the value is determined from the file format: header is set to TRUE if and only if the first row contains one fewer field than the number of columns. The default is TRUE. See read.csv.

Value

the group label vector.

Examples

```
## Not run:
data(hb)
summaryDE <- indDE(data=data,group=group,data.type="microarray",
                   case.label="2", ctrl.label="1")
pData <- summaryDE[,c(3,1)]
write.csv(pData,file="pData.csv")
pData <- read.pData(p.file="pData.csv")

## End(Not run)
```

`read.rawData` *Read raw data: microarray or RNAseq.*

Description

Read raw data: microarray & RNAseq The `read.rawData` is function to read raw data

Usage

```
read.rawData(data.file, sep = ",", quote = "\"", header = T)
```

Arguments

`data.file:`	the file name of the data to be read.
`sep:`	the field separator character. Values on each line of the file are separated by this character. If `sep = ""`, the separator is 'white space', that is one or more spaces, tabs, newlines or carriage returns. The default is `,`. See read.csv.
`quote:`	the set of quoting characters. To disable quoting altogether, use `quote = ""`. See scan for the behaviour on quotes embedded in quotes. Quoting is only considered for columns read as character, which is all of them unless colClasses is specified. The default is `\"`. See read.csv
`header:`	a logical value indicating whether the file contains the names of the variables as its first line. If missing, the value is determined from the file format: `header` is set to `TRUE` if and only if the first row contains one fewer field than the number of columns. The default is `TRUE`. See read.csv.

Value

the data matrix.

Examples

```
## Not run:
data(hb)
write.csv(data,file="rawData.csv")
data <- read.rawData(data.file="rawData.csv")

## End(Not run)
```

singleARS_global *Resemblance analysis for single pair: global ARS and its permuted p-value.*

Description

Resemblance analysis for single pair: global ARS and its permuted p-value The singleARS_global is function to perform resemblance analysis for single pair, generating global ARS and its permuted p-value.

Usage

```
singleARS_global(mcmc.merge.list, measure = "Fmeasure", parallel = F,
  cpu = 2, B = 50)
```

Arguments

mcmc.merge.list:	a list of merged MCMC output matrices.
measure:	three types of ARS measures to be used: "youden", "Fmeasure","geo.mean". Default is "Fmeasure".
parallel:	whether to perform parallel computing in permutation.
cpu:	if parallel=T, how many cpus to be used.
B:	number of permutations.

Value

global ARS values and its permuted p-value, in addition, the two values are written to the folder named "arsGlobal".

Examples

```
## Not run:
#mcmc.merge.list from the merge step
ARS_global <- singleARS_global(mcmc.merge.list,B=100)

## End(Not run)
```

singleARS_pathway *Resemblance analysis for single pair: pathway specific ARS and their permuted p-value.*

Description

Resemblance analysis for single pair: pathway specific ARS and their permuted p-value The singleARS_pathway is function to perform resemblance analysis for single pair, generating pathway specific ARS and their permuted p-value.

Usage

```
singleARS_pathway(mcmc.merge.list, select.pathway.list, measure = "Fmeasure",
  parallel = F, cpu = 2, B = 50)
```

Arguments

mcmc.merge.list:	a list of merged MCMC output matrices.
select.pathway.list:	a list of selected pathways (containing gene components).
measure:	three types of ARS measures to be used: "youden", "Fmeasure","geo.mean". Default is "Fmeasure".
parallel:	whether to perform parallel computing in permutation.
cpu:	if parallel=T, how many cpus to be used.
B:	number of permutations.

Value

a data frame of pathway specific ARS values and their permuted p-value (pathway on rows, 1st column being ARS value and 2nd column being the p-values), in addition, the dataframe is written to the folder named "arsPathway".

Examples

```
## Not run:
#mcmc.merge.list from the merge step
#select.pathway.list from the pathSelect step
ARS_pathway <- singleARS_pathway(mcmc.merge.list,select.pathway.list,B=100)

## End(Not run)
```

---

singleOutput *Analysis results for single pair: visualization outputs*

Description

Analysis results for single pair: visualization outputs for each pathway The singleOutput is function to generate visualization outputs for single pair: including heatmap of gene posterior mean, kegg pathway topology for each pathway

Usage

```
singleOutput(mcmc.merge.list, dataset.names, select.pathway.list,
  output = c("genePM", "keggView"), kegg_pathname = NULL,
  hs_gene_id = NULL)
```

Arguments

mcmc.merge.list:	a list of merged MCMC output matrices.
select.pathway.list:	a list of selected pathways (containing gene components).
dataset.names:	a vector of dataset names.
output:	two options: "genePM" (generating heatmap of gene posterior mean),"keggView" (generating kegg pathway topology, human KEGG only). cannot be empty.
kegg_pathname:	KEGG pathway name list. For "keggView" only.
hs_gene_id:	Human sapiens gene id. For "keggView" only.

Value

stored output in created folders.

Examples

```
## Not run:
#mcmc.merge.list from the merge step
#select.pathway.list from the pathSelect step
dataset.names = c("hb","mb")
library(KEGG.db)
kegg_pathname <- unlist(as.list(KEGGPATHID2NAME))
library("org.Hs.eg.db")
hs_gene_id <- unlist(mget(x=rownames(mcmc.merge.list[[1]]),
envir=org.Hs.egALIAS2EG))
singleOutput(mcmc.merge.list,dataset.names,select.pathway.list,
output=c("genePM","keggView"))

## End(Not run)
```

Index