Package ‘CoGAPS’

April 25, 2017

Version 2.10.0
Date 2014-08-23
Title Coordinated Gene Activity in Pattern Sets
Author Wai-shing Lee, Conor Kelton, Ondrej Maxian, Jacob Carey, Genevieve Stein-O’Brien, Michael Considine, John Stansfield, Shawn Sivy, Carlo Colantuoni, Alexander Favorov, Mike Ochs, Elana Fertig
Description Coordinated Gene Activity in Pattern Sets (CoGAPS) implements a Bayesian MCMC matrix factorization algorithm, GAPS, and links it to gene set statistic methods to infer biological process activity. It can be used to perform sparse matrix factorization on any data, and when this data represents biomolecules, to do gene set analysis.
Maintainer Elana J. Fertig <ejfertig@jhmi.edu>
Depends R (>= 3.0.1), Rcpp (>= 0.11.2)
Imports RColorBrewer (>= 1.0.5), gplots (>= 2.8.0), graphics, grDevices, methods, cluster, shiny, stats, utils, doParallel, foreach, ggplot2, reshape2
Suggests testthat, iterators, parallel, lintr
LinkingTo Rcpp, BH, RcppArmadillo
License GPL (==2)
biocViews GeneExpression, Transcription, GeneSetEnrichment, DifferentialExpression, Bayesian, Clustering, TimeCourse, RNASeq, Microarray, MultipleComparison, DimensionReduction
NeedsCompilation yes
RoxygenNote 5.0.1

R topics documented:

CoGAPS-package .......................................................... 2
binaryA ................................................................. 3
calcCoGAPSSStat ....................................................... 3
calcGeneGSStat ......................................................... 4
calcZ ................................................................. 4
CoGAPS ............................................................. 5
computeGeneGSProb .................................................. 6
CoGAPS-package

CoGAPS: Coordinated Gene Activity in Pattern Sets

Description

CoGAPS implements a Bayesian MCMC matrix factorization algorithm, GAPS, and links it to gene set statistic methods to infer biological process activity. It can be used to perform sparse matrix factorization on any data, and when this data represents biomolecules, to do gene set analysis.

Index

CoGAPS-package

Author(s)

Maintainer: Elana J. Fertig <ejfertig@jhmi.edu>, Michael F. Ochs <ochsm@tcnj.edu>
### References


---

### binaryA

`binaryA` creates a binarized heatmap of the A matrix in which the value is 1 if the value in Amean is greater than threshold * Asd and 0 otherwise.

**Description**

`binaryA` creates a binarized heatmap of the A matrix in which the value is 1 if the value in Amean is greater than threshold * Asd and 0 otherwise.

**Usage**

```
binaryA(Amean, Asd, threshold = 3)
```

**Arguments**

- **Amean**: the mean estimate for the A matrix
- **Asd**: the standard deviations on Amean
- **threshold**: the number of standard deviations above zero that an element of Amean must be to get a value of 1

---

### calcCoGAPSStat

`calcCoGAPSStat` calculates the gene set statistics for each column of A using a Z-score from the elements of the A matrix, the input gene set, and permutation tests.

**Description**

`calcCoGAPSStat` calculates the gene set statistics for each column of A using a Z-score from the elements of the A matrix, the input gene set, and permutation tests.

**Usage**

```
calcCoGAPSStat(Amean, Asd, GStoGenes, numPerm = 500)
```

**Arguments**

- **Amean**: A matrix mean values
- **Asd**: A matrix standard deviations
- **GStoGenes**: data.frame or list with gene sets
- **numPerm**: number of permutations for null
calcGeneGSStat

calcGeneGSStat calculates the probability that a gene listed in a gene set behaves like other genes in the set within the given data set.

Usage

calcGeneGSStat(Amean, Asd, GSGenes, numPerm, Pw = rep(1, ncol(Amean)), nullGenes = F)

Arguments

- Amean: A matrix mean values
- Asd: A matrix standard deviations
- GSGenes: data.frame or list with gene sets
- numPerm: number of permutations for null
- Pw: weight on genes
- nullGenes: logical indicating gene adjustment

---

calcZ

calcZ calculates the Z-score for each element based on input mean and standard deviation matrices.

Usage

calcZ(meanMat, sdMat)

Arguments

- meanMat: matrix of mean values
- sdMat: matrix of standard deviation values
CoGAPS calls the C++ MCMC code through gapsRun and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix and then calls calcCoGAPSSStat to estimate gene set activity with nPerm set to 500.

**Description**

CoGAPS calls the C++ MCMC code through gapsRun and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix and then calls calcCoGAPSSStat to estimate gene set activity with nPerm set to 500.

**Usage**

```r
CoGAPS(data, unc, ABins = data.frame(), PBins = data.frame(), GStoGenes, nFactor = 7, simulation_id = "simulation", nEquil = 1000, nSample = 1000, nOutR = 1000, output_atomic = FALSE, fixedBinProbs = FALSE, fixedDomain = "N", sampleSnapshots = TRUE, numSnapshots = 100, plot = TRUE, nPerm = 500, alphaA = 0.01, nMaxA = 1e+05, max_gibbmass_paraA = 100, alphaP = 0.01, nMaxP = 1e+05, max_gibbmass_paraP = 100)
```

**Arguments**

- `data`: data matrix
- `unc`: uncertainty matrix (std devs for chi-squared of Log Likelihood)
- `ABins`: a matrix of same size as A which gives relative probability of that element being non-zero
- `PBins`: a matrix of same size as P which gives relative probability of that element being non-zero
- `GStoGenes`: data.frame or list with gene sets
- `nFactor`: number of patterns (basis vectors, metagenes)
- `simulation_id`: name to attach to atoms files if created
- `nEquil`: number of iterations for burn-in
- `nSample`: number of iterations for sampling
- `nOutR`: how often to print status into R by iterations
- `output_atomic`: whether to write atom files (large)
- `fixedBinProbs`: Boolean for using relative probabilities given in Abins and Pbins
- `fixedDomain`: character to indicate whether A or P is domain for relative probabilities
- `sampleSnapshots`: Boolean to indicate whether to capture individual samples from Markov chain during sampling
- `numSnapshots`: the number of individual samples to capture
- `plot`: Boolean to indicate whether to produce output graphics
- `nPerm`: number of permutations in gene set test
- `alphaA`: sparsity parameter for A domain
computeGeneGSProb

Description
Computes the p-value for gene set membership using the CoGAPS-based statistics developed in Fertig et al. (2012). This statistic refines set membership for each candidate gene in a set specified in GSGenes by comparing the inferred activity of that gene to the average activity of the set. Specifically, we compute the following summary statistic for each gene \( g \) that is a candidate member of gene set \( G \):

\[
S_{g,G} = \frac{\sum_{p} -\log(Pr_{G,p})Pw[p](A_{gp}/\sigma_{gp})}{\sum_{p} -\log(Pr_{G,p})Pw[p]},
\]

where \( p \) indexes each of the patterns, \( Pr_{G,p} \) is the probability that gene set \( G \) is upregulated computed with \texttt{calcCoGAPSStat}, \( A_{gp} \) is the mean amplitude matrix from the GAPS matrix factorization, \( Pw[p] \) is a prior weighting for each pattern based upon the context to which that pattern relates, and \( \sigma_{gp} \) is the standard deviation of the amplitude matrix. P-values are formulated from a permutation test comparing the value of \( S_{g,G} \) for genes in GSGenes relative to the value of \( S_{g,G} \) numPerm random gene sets with the same number of targets.

Usage
\texttt{computeGeneGSProb(Amean, Asd, GSGenes, Pw=rep(1,ncol(Amean)),numPerm=500,PwNull=F)}

Arguments
- \texttt{Amean} Sampled mean value of the amplitude matrix \( A \). row.names(Amean) must correspond to the gene names contained in GSGenes.
- \texttt{Asd} Sampled standard deviation of the amplitude matrix \( A \).
- \texttt{GSGenes} Vector containing the prior estimate of members of the gene set of interest.
- \texttt{Pw} Vector containing the weight to assign each pattern in the gene statistic assumed to be computed from the association of the pattern with samples in a given context (optional: default=1 giving all patterns equal weight).
- \texttt{numPerm} Number of permutations used for the null distribution in the gene set statistic. (optional; default=500)
- \texttt{PwNull} Logical value. If TRUE, use pattern weighting in Pw when computing the null distribution for the statistic. If FALSE, do not use the pattern weighting so that the null is context independent. (optional; default=F)
Value

A vector of length GSGenes containing the p-values of set membership for each gene contained in the set specified in GSGenes.

Author(s)

Elana J. Fertig <ejfertig@jhmi.edu>

References


See Also

calcCoGAPStat

Examples

## Not run:

#########################################################################
# Results for GIST data in Fertig et al. (2012) #
#########################################################################

# load the data
data('GIST_TS_20084')
data('TFGSList')

# define transcription factors of interest based on Ochs et al. (2009)

# run the GAPS matrix factorization
nIter <- 10000
results <- CoGAPS(GIST.D, GIST.S, tf2ugFC,
    nFactor=5,
    nEquil=nIter, nSample=nIter,
    plot=FALSE)

# set membership statistics
permTFStats <- list()
for (tf in TFs) {
    genes <- levels(tf2ugFC[,tf])
    genes <- genes[2:length(genes)]
    permTFStats[[tf]] <- computeGeneGSProb(Amean = GISTResults$Amean,
        Asd = GISTResults$Asd, genes)
}

## End(Not run)
createGWCoGAPSSets

Description

createGWCoGAPSSets factors whole genome data into randomly generated sets for indexing;

Usage

createGWCoGAPSSets(data = D, nSets = nSets, 
outRDA = "GenesInCoGAPSSets.Rda", keep = TRUE)

Arguments

data data matrix with unique rownames
nSets number of sets for parallelization
outRDA name of output file
keep logical indicating whether or not to save gene set list. Default is TRUE.

Value

list with randomly generated sets of genes from whole genome data

Examples

## Not run:
createGWCoGAPSet(D,nSets=nSets)
## End(Not run)

gapsMapRun
gapsMapRun calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix; as opposed to gapsRun, this method takes an additional input specifying set patterns in the P matrix

Description

gapsMapRun calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix; as opposed to gapsRun, this method takes an additional input specifying set patterns in the P matrix

Usage

gapsMapRun(D, S, FP, ABins = data.frame(), PBins = data.frame(), 
nFactor = 5, simulation_id = "simulation", nEquil = 1000, 
nSample = 1000, nOutR = 1000, output_atomic = FALSE, 
fixedMatrix = "P", fixedBinProbs = FALSE, fixedDomain = "N", 
sampleSnapshots = TRUE, numSnapshots = 100, alphaA = 0.01, 
nMaxA = 1e+05, max_gibbmass_paraA = 100, alphaP = 0.01, nMaxP = 1e+05, 
max_gibbmass_paraP = 100, seed = -1, messages = TRUE)
Arguments

D  data matrix
S  uncertainty matrix (std devs for chi-squared of Log Likelihood)
FP  data.frame with rows giving fixed patterns for P
ABins  a matrix of same size as A which gives relative probability of that element being non-zero
PBins  a matrix of same size as P which gives relative probability of that element being non-zero
nFactor  number of patterns (basis vectors, metagenes), which must be greater than or equal to the number of rows of FP
simulation_id  name to attach to atoms files if created
nEquil  number of iterations for burn-in
nSample  number of iterations for sampling
nOutR  how often to print status into R by iterations
output_atomic  whether to write atom files (large)
fixedMatrix  character indicating whether A or P matrix has fixed columns or rows respectively
fixedBinProbs  Boolean for using relative probabilities given in Abins and Pbins
fixedDomain  character to indicate whether A or P is domain for relative probabilities
sampleSnapshots  Boolean to indicate whether to capture individual samples from Markov chain during sampling
numSnapshots  the number of individual samples to capture
alphaA  sparsity parameter for A domain
nMaxA  PRESENTLY UNUSED, future = limit number of atoms
max_gibbmass_paraA  limit truncated normal to max size
alphaP  sparsity parameter for P domain
nMaxP  PRESENTLY UNUSED, future = limit number of atoms
max_gibbmass_paraP  limit truncated normal to max size
seed  Set seed for reproducibility. Positive values provide initial seed, negative values just use the time.
messages  Display progress messages
gapsMapTestRun calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix; as opposed to gapsRun, this method takes an additional input specifying set patterns in the P matrix.

**Description**

GapsMapTestRun calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix; as opposed to gapsRun, this method takes an additional input specifying set patterns in the P matrix.

**Usage**

```r
gapsMapTestRun(D, S, FP, ABins = data.frame(), PBins = data.frame(), nFactor = 7, simulation_id = "simulation", nEquil = 1000, nSample = 1000, nOutR = 1000, output_atomic = FALSE, fixedMatrix = "P", fixedBinProbs = FALSE, fixedDomain = "N", alphaA = 0.01, nMaxA = 1e+05, max_gibbmass_paraA = 100, alphaP = 0.01, nMaxP = 1e+05, max_gibbmass_paraP = 100)
```

**Arguments**

- **D**  
  data matrix

- **S**  
  uncertainty matrix (std devs for chi-squared of Log Likelihood)

- **FP**  
  data.frame with rows giving fixed patterns for P

- **ABins**  
  a matrix of same size as A which gives relative probability of that element being non-zero

- **PBins**  
  a matrix of same size as P which gives relative probability of that element being non-zero

- **nFactor**  
  number of patterns (basis vectors, metagenes), which must be greater than or equal to the number of rows of FP

- **simulation_id**  
  name to attach to atoms files if created

- **nEquil**  
  number of iterations for burn-in

- **nSample**  
  number of iterations for sampling

- **nOutR**  
  how often to print status into R by iterations

- **output_atomic**  
  whether to write atom files (large)

- **fixedMatrix**  
  character indicating whether A or P matrix has fixed columns or rows respectively

- **fixedBinProbs**  
  Boolean for using relative probabilities given in Abins and Pbins

- **fixedDomain**  
  character to indicate whether A or P is domain for relative probabilities

- **alphaA**  
  sparsity parameter for A domain

- **nMaxA**  
  PRESENTLY UNUSED, future = limit number of atoms

- **max_gibbmass_paraA**  
  limit truncated normal to max size

- **alphaP**  
  sparsity parameter for P domain
gapsRun

gapsRun calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix.

Description

gapsRun calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix.

Usage

gapsRun(D, S, ABins = data.frame(), PBins = data.frame(), nFactor = 7, simulation_id = "simulation", nEquil = 1000, nSample = 1000, nOutR = 1000, output_atomic = FALSE, fixedBinProbs = FALSE, fixedDomain = "N", sampleSnapshots = TRUE, numSnapshots = 100, alphaA = 0.01, nMaxA = 1e+05, max_gibbmass_paraA = 100, alphaP = 0.01, nMaxP = 1e+05, max_gibbmass_paraP = 100, seed = -1, messages = TRUE)

Arguments

D

data matrix

S

uncertainty matrix (std devs for chi-squared of Log Likelihood)

ABins

a matrix of same size as A which gives relative probability of that element being non-zero

PBins

a matrix of same size as P which gives relative probability of that element being non-zero

nFactor

number of patterns (basis vectors, metagenes), which must be greater than or equal to the number of rows of FP

simulation_id

name to attach to atoms files if created

nEquil

number of iterations for burn-in

nSample

number of iterations for sampling

nOutR

how often to print status into R by iterations

output_atomic

whether to write atom files (large)

fixedBinProbs

Boolean for using relative probabilities given in Abins and Pbins

fixedDomain

character to indicate whether A or P is domain for relative probabilities

sampleSnapshots

Boolean to indicate whether to capture individual samples from Markov chain during sampling

numSnapshots

the number of individual samples to capture

alphaA

sparsity parameter for A domain

nMaxA

PRESENTLY UNUSED, future = limit number of atoms

max_gibbmass_paraA

limit truncated normal to max size
gapsTestRun calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix

Description

gapsTestRun calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix

Usage

gapsTestRun(D, S, ABins = data.frame(), PBins = data.frame(), nFactor = 7, simulation_id = "simulation", nEquil = 1000, nSample = 1000, nOutR = 1000, output.atomic = FALSE, fixedBinProbs = FALSE, fixedDomain = "N", alphaA = 0.01, nMaxA = 1e+05, max_gibbmass_paraA = 100, alphaP = 0.01, nMaxP = 1e+05, max_gibbmass_paraP = 100)

Arguments

D data matrix
S uncertainty matrix (std devs for chi-squared of Log Likelihood)
ABins a matrix of same size as A which gives relative probability of that element being non-zero
PBins a matrix of same size as P which gives relative probability of that element being non-zero
nFactor number of patterns (basis vectors, metagenes), which must be greater than or equal to the number of rows of FP
simulation_id name to attach to atoms files if created
nEquil number of iterations for burn-in
nSample number of iterations for sampling
nOutR how often to print status into R by iterations
output_atomic whether to write atom files (large)
fixedBinProbs Boolean for using relative probabilities given in Abins and Pbins
fixedDomain character to indicate whether A or P is domain for relative probabilities
**generateSeeds**

alphaA  
sparsity parameter for A domain

nMaxA  
PRESENTLY UNUSED, future = limit number of atoms

max_gibbmass_paraA  
limit truncated normal to max size

alphaP  
sparsity parameter for P domain

nMaxP  
PRESENTLY UNUSED, future = limit number of atoms

max_gibbmass_paraP  
limit truncated normal to max size

---

**generateSeeds**  
**generateSeeds**

---

**Description**

generateSeeds

**Usage**

generateSeeds(chains = 2, seed = -1)

**Arguments**

chains  
number of MCMC chains to be used

seed  
numeric indicating whether to generate seed from system clock. Default is -1.

**Value**

vector of randomly generated seeds for use with gapsRun, gapsMapRun, or GWCoGAPS

**Examples**

## Not run:
generateSeeds(chains=2, seed=-1)

## End(Not run)
Sample GIST gene expression data from Ochs et al. (2009).

**Description**

Gene expression data from gastrointestinal stromal tumor cell lines treated with Gleevec.

**Usage**

GIST_TS_20084

**Format**

Matrix with 1363 genes by 9 samples of mean gene expression data.

**References**


Sample GIST gene expression data from Ochs et al. (2009).

**Description**

Standard deviation of gene expression data from gastrointestinal stromal tumor cell lines treated with Gleevec.

**Usage**

GIST_TS_20084

**Format**

Matrix with 1363 genes by 9 samples containing standard deviation (GIST.S) of the gene expression data.

**References**

GSets

Simulated dataset to quantify gene set membership.

Description

Simulated gene sets used to generate amplitude matrix in SimpSim.A and corresponding data SimpSim.D.

Usage

GSets

Format

A list containing names of genes in two simulated gene sets used to generate the data in SimpSim.D.

GWCoGAPS

GWCoGAPS

Description

GWCoGAPS calls the C++ MCMC code and performs Bayesian matrix factorization returning the two matrices that reconstruct the data matrix for whole genome data;

Usage

GWCoGAPS(D, S, nFactor, nSets, nCores = NA, saveBySetResults = FALSE, fname = "GWCoGAPS.AP.fixed", PatternsMatchFN = patternMatch4Parallel, Cut = NA, minNS = NA, ...)  

Arguments

D data matrix
S uncertainty matrix (std devs for chi-squared of Log Likelihood)
nFactor number of patterns (basis vectors, metagenes), which must be greater than or equal to the number of rows of FP
nSets number of sets for parallelization
nCores number of cores for parallelization. If left to the default NA, nCores = nSets.
saveBySetResults logical indicating whether to save by intermediary by set results. Default is FALSE.
fname character string used to label file output. Default is "GWCoGAPS.AP.fixed"
PatternsMatchFN function to use for pattern matching across sets
Cut number of branches at which to cut dendrogram used in patternMatch4Parallel
minNS minimum of individual set contributions a cluster must contain
... additional parameters to be fed into gapsRun and gapsMapRun
patternMarkers

Description
patternMarkers

Usage
patternMarkers(Amatrix = NA, scaledPmatrix = FALSE, Pmatrix = NA, threshold = "All", lp = NA, full = FALSE)

Arguments
- **Amatrix**: A matrix of genes by weights resulting from CoGAPS or other NMF decomposition
- **scaledPmatrix**: logical indicating whether the corresponding pattern matrix was fixed to have max 1 during decomposition
- **Pmatrix**: the corresponding Pmatrix (patterns X samples) for the provided Amatrix (genes x patterns). This must be supplied if scaledPmatrix is FALSE.
- **threshold**: the type of threshold to be used. The default "All" will distribute genes into pattern with the lowest ranking. The "cut" thresholding by the first gene to have a lower ranking, i.e. better fit to, a pattern.
- **lp**: a vector of weights for each pattern to be used for finding markers. If NA markers for each pattern of the A matrix will be used.
- **full**: logical indicating whether to return the ranks of each gene for each pattern

Value
By default a non-overlapping list of genes associated with each lp. If full=TRUE a data.frame of genes rankings with a column for each lp will also be returned.

Examples
```r
## Not run:
patternMarkers(Amatrix=AP$Amean,scaledPmatrix=FALSE,Pmatrix=NA,threshold="cut")
## End(Not run)
```
patternMatch4Parallel

Description

patternMatch4Parallel

Usage

patternMatch4Parallel(Ptot, nSets, cnt, minNS, cluster.method = "complete", ignore.NA = FALSE, bySet = FALSE, ...)

Arguments

- **Ptot**: a matrix containing the total by set estimates of Pmean output from `reOrderBySet`
- **nSets**: number of parallel sets used to generate `Ptot`
- **cnt**: number of branches at which to cut dendrogram
- **minNS**: minimum of individual set contributions a cluster must contain
- **cluster.method**: the agglomeration method to be used for clustering
- **ignore.NA**: logical indicating whether or not to ignore NAs from potential over dimensionalization. Default is FALSE.
- **bySet**: logical indicating whether to return list of matched set solutions from `Ptot`
- **...**: additional parameters for `agnes`

Value

- a matrix of concensus patterns by samples. If `bySet=TRUE` then a list of the set contributions to each concensus pattern is also returned.

See Also

- `agnes`

PatternMatcher Shiny Ap

Description

PatternMatcher Shiny Ap

Usage

patternMatcher(PBySet = NULL, out = NULL, order = NULL, sample.color = NULL)
Arguments

PBySet: list of matched set solutions for the Pmatrix from an NMF algorithm
out: optional name for saving output
order: optional vector indicating order of samples for plotting. Default is NULL.
sample.color: optional vector of colors of same length as colnames. Default is NULL.

Value

either an index of selected sets’ contributions or the edited PBySet object

Examples

## Not run:
patternMatcher(PBySet,out,order,sample.color)
## End(Not run)

plotAtoms

plotAtoms a simple plot of the number of atoms from one of the vectors returned with atom numbers

Description

plotAtoms a simple plot of the number of atoms from one of the vectors returned with atom numbers

Usage

plotAtoms(gapsRes, type = "sampA")

Arguments

gapsRes: the list resulting from applying GAPS
type: the atoms to plot, values are "sampA", "sampP", "equilA", or "equilP" to plot sampling or equilibration topology atom numbers

plotDiag

plotDiag plots a series of diagnostic plots

Description

plotDiag plots a series of diagnostic plots

Usage

plotDiag(gapsRes)

Arguments

gapsRes: list returned by gapsRun, gapsMapRun, or CoGAPS
plotGAPS

**Description**

plotGAPS plots the output A and P matrices as a heatmap and line plot respectively

**Usage**

plotGAPS(A, P, outputPDF = "")

**Arguments**

- A: the mean A matrix
- P: the mean P matrix
- outputPDF: optional root name for PDF output, if not specified, output goes to screen

---

plotP

**Description**

plotP plots the P matrix in a line plot with error bars

**Usage**

plotP(PMean_Mat, P_SD)

**Arguments**

- PMean_Mat: matrix of mean values of P
- P_SD: matrix of standard deviation values of P

---

plotPatternMarkers

**Description**

plotPatternMarkers

**Usage**

plotPatternMarkers(data = NA, patternMarkers = NA, patternPalette = NA, sampleNames = NA, samplePalette = NULL, colDenogram = TRUE, heatmapCol = "bluered", scale = "row", ...)
plotSmoothPatterns

Arguments

data the dataset from which the patterns were generated
patternMarkers the list of genes generated from the patternMarkers function
patternPalette a vector indicating what color should be used for each pattern
sampleNames names of the samples to use for labeling
samplePalette a vector indicating what color should be used for each sample
colDenogram logical indicating whether to display sample denogram
heatmapCol palette giving color scheme for heatmap
scale character indicating if the values should be centered and scaled in either the row direction or the column direction, or none. The default is "row".
... additional graphical parameters to be passed to heatmap.2

Value

heatmap of the data values for the patternMarkers

See Also

heatmap.2

Examples

## Not run:
plotPatternMarkers(data=p,patternMarkers=PatternMarkers,patternPalette=NULL,sampleNames=pd$sample,
samplePalette=pd$color,colDenogram=TRUE,heatmapCol="bluered", scale='row')
## End(Not run)

plotSmoothPatterns plots the output A and P matrices as a heatmap and line plot respectively

Description

plotSmoothPatterns plots the output A and P matrices as a heatmap and line plot respectively

Usage

plotSmoothPatterns(P, x = NULL, breaks = NULL, breakStyle = T,
orderP = !all(is.null(x)), plotPTS = F, pointCol = "black",
lineCol = "grey", add = F, ...)
Arguments

- **P**: the mean A matrix
- **x**: optional variables
- **breaks**: breaks in plots
- **breakStyle**: style of breaks
- **orderP**: whether to order patterns
- **plotPTS**: whether to plot points on lines
- **pointCol**: color of points
- **lineCol**: color of line
- **add**: logical specifying if bars should be added to an already existing plot; defaults to ‘FALSE’.
- **...**: arguments to be passed to/from other methods. For the default method these can include further arguments (such as ‘axes’, ‘asp’ and ‘main’) and graphical parameters (see ‘par’) which are passed to ‘plot.window()’, ‘title()’ and ‘axis’.

Description

postFixed4Parallel

Usage

postFixed4Parallel(AP.fixed = NA, setPs = NA)

Arguments

- **AP.fixed**: output of parallel gapsMapRun calls with same FP
- **setPs**: data.frame with rows giving fixed patterns for P used as input for gapsMapRun

Value

list of two data.frames containing the A matrix estimates or their corresponding standard deviations from output of parallel gapsMapRun
reconstructGene

Description
reconstruct Gene

Usage
reconstructGene(A = NA, P = NA, genes = NA)

Arguments
- **A**: A matrix estimates
- **P**: corresponding P matrix estimates
- **genes**: an index of the gene or genes of interest. If NA, the default, all genes contained in A will be returned.

Value
the $D'$ estimate of a gene or set of genes

reorderByPatternMatch

Description
reorderByPatternMatch plots the output A and P matrices as a heatmap and line plot respectively

Usage
reorderByPatternMatch(P, matchTo)

Arguments
- **P**: matrix to be matched
- **matchTo**: matrix to match P to
**reOrderBySet**

**Description**

&lt;restructures output of gapsRun into a list containing each sets solution for Amean, Pmean, and Asd&gt;

**Usage**

`reOrderBySet(AP, nFactor, nSets)`

**Arguments**

- **AP**  
  output of gapsRun in parallel
- **nFactor**  
  number of patterns
- **nSets**  
  number of sets

**Value**

a list containing the `nSets` sets solution for Amean under "A", Pmean under "P", and Asd under "Asd"

**Examples**

```r
## Not run:
reOrderBySet(AP,nFactor,nSets)
## End(Not run)
```

**residuals**

**Description**

residuals calculate residuals and produce heatmap

**Usage**

`residuals(AMean_Mat, PMean_Mat, D, S)`

**Arguments**

- **AMean_Mat**  
  matrix of mean values for A from GAPS
- **PMean_Mat**  
  matrix of mean values for P from GAPS
- **D**  
  original data matrix run through GAPS
- **S**  
  original standard deviation matrix run through GAPS
SimpSim.A

**Description**
True amplitude matrix generated from gene sets in GSets used to generate simulated data in SimpSim.D.

**Usage**
SimpSim.A

**Format**
Matrix with 30 genes by 3 patterns of true amplitude used to generate simulated data.

SimpSim.D

**Description**
Simulated gene expression data from true patterns in SimpSim.P and amplitude in SimpSim.A.

**Usage**
SimpSim.D

**Format**
Matrix with 30 genes by 20 samples of simulated gene expression data.

SimpSim.P

**Description**
True pattern matrix used to generate simulated data in SimpSim.D.

**Usage**
SimpSim.P

**Format**
Matrix with 3 patterns by 20 samples of true patterns used to generate simulated data.
SimpSim.S

**Description**
Standard deviation of simulated gene expression data from true patterns in SimpSim.P and amplitude in SimpSim.A.

**Usage**
SimpSim.S

**Format**
Matrix with 30 genes by 20 samples of containing standard deviation of simulated gene expression data.

tf2ugFC

**Description**
List of genes contained in gastrointestinal stromal tumor cell line measurements that are regulated by transcription factors in the TRANSFAC database. Used for the gene set analysis in Ochs et al. (2009).

**Usage**
TFGSList

**Format**
Data.frame containing genes (rows) regulated by each transcription factor (columns).

**References**
Index

*Topic datasets
  GIST.D, 14
  GIST.S, 14
  GSets, 15
  SimpSim.A, 24
  SimpSim.D, 24
  SimpSim.P, 24
  SimpSim.S, 25
  tf2ugFC, 25

*Topic misc
  computeGeneGSProb, 6

agnes, 17

binaryA, 3
calcCoGAPSSStat, 3, 6, 7
calcGeneGSStat, 4
calcZ, 4
CoGAPS, 5
CoGAPS-package, 2
computeGeneGSProb, 6
createGWCoGAPSSets, 8
gapsMapRun, 8, 16
gapsMapTestRun, 10
gapsRun, 11, 16
gapsTestRun, 12
geneGSProb (computeGeneGSProb), 6
generateSeeds, 13
GIST.D, 14
GIST.S, 14
GSets, 15, 24
GWCoGAPS, 15

heatmap.2, 20

list, 15

patternMarkers, 16
patternMatch4Parallel, 16, 17
patternMatcher, 17
plotAtoms, 18
plotDiag, 18
plotGAPS, 19
plotP, 19
plotPatternMarkers, 19
plotSmoothPatterns, 20
postFixed4Parallel, 21
reconstructGene, 22
reorderByPatternMatch, 22
reOrderBySet, 23
residuals, 23
SimpSim.A, 15, 24, 25
SimpSim.D, 15, 24
SimpSim.P, 24, 24, 25
SimpSim.S, 25
tf2ugFC, 25