Package 'CoGAPS'

April 9, 2015

Version 2.0.0

Date 2014-08-23

Title Coordinated Gene Activity in Pattern Sets

Author Elana J. Fertig, Michael F. Ochs

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>, Michael F. Ochs <ochsm@tcnj.edu>

Depends R (>= 3.0.1), Rcpp (>= 0.11.2), RColorBrewer (>= 1.0.5), gplots (>= 2.8.0)

Imports graphics, grDevices, methods, stats, utils

LinkingTo Rcpp, BH

License GPL (==2)

biocViews GeneExpression, Microarray

R topics documented:

`CoGAPS-package` *CoGAPS: Coordinated Gene Analysis 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.

Package:	CoGAPS
Type:	Package
Version:	1.0
Date:	2014-07-23
License:	LGPL

Author(s)

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

References

Fertig EJ, Ding J, Favorov AV, Parmigiani G, Ochs MF. CoGAPS: an R/C++ package to identify patterns and biological process activity in transcriptomic data. Bioinformatics. 2010 Nov 1;26(21):2792-3

`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 *CoGAPS gene set statistic*

Description

Computes the p-value for the association of underlying patterns from microarray data to activity in gene sets.

Usage

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

Arguments

Amean	Sampled mean value of the amplitude matrix A. row.names(Amean) must correspond to the gene names contained in GStoGenes.
Asd	Sampled standard deviation of the amplitude matrix A.
GStoGenes	List or data frame containing the genes in each gene set. If a list, gene set names are the list names and corresponding elements are the names of genes contained in each set. If a data frame, gene set names are in the first column and corresponding gene names are listed in rows beneath each gene set name.
numPerm	Number of permuations used for the null distribution in the gene set statistic. (optional; default=500)

Details

This script links the patterns identified in the columns of P to activity in each of the gene sets specified in GStoGenes using a novel z-score based statistic developed in Ochs et al. (2009). Specifically, the z-score for pattern p and gene set G_i containing G total genes is given by

$$Z_{i,p} = \frac{1}{G} \sum_{g in G_i} A_{gp}/\sigma_{gp}$$

, where g indexes the genes in the set and σ_{gp} is the standard deviation of A_{gp} obtained from MCMC sampling. CoGAPS then uses the specified `numPerm` random sample tests to compute a consistent p value estimate from that z score.

Value

A list containing:

`GSUpreg`	p-values for upregulation of each gene set in each pattern.
`GSDownreg`	p-values for downregulation of each gene set in each pattern.
`GSActEst`	p-values for activity of each gene set in each pattern.

Author(s)

Elana J. Fertig <ejfertig@jhmi.edu>

References

M.F. Ochs, L. Rink, C. Tarn, S. Mburu, T. Taguchi, B. Eisenberg, and A.K. Godwin. (2009) Detection and treatment-induced changes in signaling pathways in gastrointestinal stromal tumors using transcriptomic data. Cancer Research, 69:9125-9132.

See Also

CoGAPS

`CoGAPS` *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 calcCoGAPSStat 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 calcCoGAPSStat to estimate gene set activity.

Usage

```
CoGAPS(data, unc, GStoGenes, nFactor = "7", nEquil = 1000, nSample = 1000,
  nOutR = 1000, output_atomic = "false", simulation_id = "simulation",
  plot = TRUE, nPerm = 500, alphaA = "0.01", nMaxA = "100000",
  max_gibbmass_paraA = "100.0", lambdaA_scale_factor = "1.0",
  alphaP = "0.01", nMaxP = "100000", max_gibbmass_paraP = "100.0",
  lambdaP_scale_factor = "1.0")
```

Arguments

data	data matrix
unc	uncertainty matrix (std devs for chi-squared of Log Likelihood)
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
plot	logical to determine if plots produced
nPerm	number of permutations for gene set test
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)
alphaA	sparsity parameter for A domain
alphaP	sparsity parameter for P domain
max_gibbmass_paraA	limit truncated normal to max size for A
max_gibbmass_paraP	limit truncated normal to max size for P
nMaxA	PRESENTLY UNUSED, future = limit number of atoms for A
nMaxP	PRESENTLY UNUSED, future = limit number of atoms for P
lambdaA_scale_factor	lambda factor in penalized likelihood for A
lambdaP_scale_factor	lambda factor in penalized likelihood for P

Details

CoGAPS first decomposes the data matrix using GAPS, $\mathbf{D}$, into a basis of underlying patterns and then determines the gene set activity in each of these patterns.

The GAPS decomposition is achieved by finding amplitude and pattern matrices ($\mathbf{A}$ and $\mathbf{P}$, respectively) for which

$$\mathbf{D} = \mathbf{AP} + \Sigma,$$

where Σ is the matrix of uncertainties given by unc. The matrices $\mathbf{A}$ and $\mathbf{P}$ are assumed to have the atomic prior described in Sibisi and Skilling (1997) and are found with MCMC sampling.

Then, the patterns identified in the columns of $\mathbf{P}$ are linked to activity in each of the gene sets specified in GStoGenes using a novel z-score based statistic developed in Ochs et al. (2009). Specifically, the z-score for pattern p and gene set G_i containing \$G\$ total genes is given by

$$Z_{i,p} = \frac{1}{G} \sum_{g in \mathcal{G}_i} \frac{\mathbf{A_{gp}}}{Asd_{gp}},$$

where g indexes the genes in the set and Asd_{gp} is the standard deviation of $\mathbf{A}_{gp}$ obtained from MCMC sampling. CoGAPS then uses the specified `nPerm` random sample tests to compute a consistent p value estimate from that z score. Note that the data from Ochs et al. (2009) are provided with this package in GIST_TS_20084.RData and TFGSList.RData are also provided with this package for further validation.

Value

A list containing:

`Amean`	Sampled mean value of the amplitude matrix $\mathbf{A}$.
`Asd`	Sampled standard deviation of the amplitude matrix $\mathbf{A}$.
`Pmean`	Sampled mean value of the amplitude matrix $\mathbf{P}$.
`Psd`	Sampled standard deviation of the amplitude matrix $\mathbf{P}$.
`D`	Data matrix $\mathbf{D}$ input to factorization.
`Sigma`	uncertainty matrix (std devs for chi-squared of Log Likelihood)
`GSUpreg`	p-values for upregulation of each gene set in each pattern.
`GSDownreg`	p-values for downregulation of each gene set in each pattern.
`GSActEst`	p-values for activity of each gene set in each pattern.
`atomsAEquil`	Number of atoms in $\mathbf{A}$ during each iteration of the equilibration phase.
`atomsASamp`	Number of atoms in $\mathbf{A}$ during each iteration of the sampling phase.
`atomsPEquil`	Number of atoms in $\mathbf{P}$ during each iteration of the equilibration phase.
`atomsPSamp`	Number of atoms in $\mathbf{P}$ during each iteration of the sampling phase.
`chiSqValues`	Value of chi^2 at each step during equilibration and sampling.
`meanChi2`	Value of chi^2 for Amean and Pmean.

See Also

`gapsRun`,`calcCoGAPSStat`

Examples

```
## Not run:
## Load data
nIter <- 5000

## Run GAPS matrix decomposition with gene set statistic
results <- CoGAPS(data=SimpSim.D, unc=SimpSim.S,
                  GStoGenes=GSets,
```

```
                    nFactor=3,
                    nEquil=nIter, nSample=nIter,
                    plot=FALSE)


## Plot the results
plotGAPS(results$Amean, results$Pmean, GSFigs)

## End(Not run)
```

---

computeGeneGSProb *CoGAPS gene membership statistic*

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:

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

where p indexes each of the patterns, $Pr_{G,p}$ is the probability that gene set G is upregulated computed with 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 σ_{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

```
computeGeneGSProb(Amean, Asd, GStoGenes, Pw=rep(1,ncol(Amean)),numPerm=500,PwNull=F)
```

Arguments

Amean	Sampled mean value of the amplitude matrix $\mathbf{A}$. row.names(Amean) must correspond to the gene names contained in GStoGenes.
Asd	Sampled standard deviation of the amplitude matrix $\mathbf{A}$.
GStoGenes	Vector containing the prior estimate of members of the gene set of interest.
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).
numPerm	Number of permuations used for the null distribution in the gene set statistic. (optional; default=500)
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 GStoGenes containing the p-values of set membership for each gene containined in the set specified in GStoGenes.

Author(s)

Elana J. Fertig <ejfertig@jhmi.edu>

References

E.J. Fertig, A.V. Favorov, and M.F. Ochs (2013) Identifying context-specific transcription factor targets from prior knowledge and gene expression data. 2012 IEEE Nanobiosciences.

See Also

calcCoGAPSStat

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)
TFs <- c("c.Jun", NF.kappaB, Smad4, "STAT3", "Elk.1", "c.Myc", "E2F.1",
         "AP.1", "CREB", "FOXO", "p53", "Sp1")

# 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]] <- computeGeneTFProb(Amean = GISTResults$Amean,
                                            Asd = GistResults$Asd, genes)
}


## 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, nFactor = "7", simulation_id = "simulation",
  nEquil = "1000", nSample = "1000", nOutR = 1000,
  output_atomic = "FALSE", alphaA = "0.01", nMaxA = "100000",
  max_gibbmass_paraA = "100.0", lambdaA_scale_factor = "1.0",
  alphaP = "0.01", nMaxP = "100000", max_gibbmass_paraP = "100.0",
  lambdaP_scale_factor = "1.0")
```

Arguments

D	data matrix
S	uncertainty matrix (std devs for chi-squared of Log Likelihood)
FP	matrix with rows giving fixed patterns for P
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)
alphaA	sparsity parameter for A domain
alphaP	sparsity parameter for P domain
max_gibbmass_paraA	limit truncated normal to max size in A
max_gibbmass_paraP	limit truncated normal to max size in P
nMaxA	PRESENTLY UNUSED, future = limit number of atoms in A
nMaxP	PRESENTLY UNUSED, future = limit number of atoms in P
lambdaA_scale_factor	lambda factor in penalized likelihood in A
lambdaP_scale_factor	lambda factor in penalized likelihood in P

Details

The decomposition in GAPS is achieved by finding amplitude and pattern matrices ($\mathbf{A}$ and $\mathbf{P}$, respectively) for which

$$\mathbf{D} = \mathbf{AP} + \Sigma$$

, where Σ is the matrix of uncertainties given by S. The matrices $\mathbf{A}$ and $\mathbf{P}$ are assumed to have the atomic prior described in Sibisi and Skilling (1997) and are found with MCMC sampling. However, some rows of $\mathbf{P}$ are fixed to be the values specified in the input argument FP after rescaling to have norm 1.

Value

A list containing:

Amean	Sampled mean value of the amplitude matrix $\mathbf{A}$.
Asd	Sampled standard deviation of the amplitude matrix $\mathbf{A}$.
Pmean	Sampled mean value of the amplitude matrix $\mathbf{P}$.
Psd	Sampled standard deviation of the amplitude matrix $\mathbf{P}$.
atomsAEquil	Number of atoms in $\mathbf{A}$ during each iteration of the equilibration phase.
atomsASamp	Number of atoms in $\mathbf{A}$ during each iteration of the sampling phase.
atomsPEquil	Number of atoms in $\mathbf{P}$ during each iteration of the equilibration phase.
atomsPSamp	Number of atoms in $\mathbf{P}$ during each iteration of the sampling phase.
chiSqValues	Value of chi^2 at each step during equilibration and sampling.
meanChi2	Value of chi^2 for Amean and Pmean.

See Also

CoGAPS,gapsRun

Examples

```
## Not run:
## Load data
data(SimpSim)

## Specify the fixed pattern
mapP <- matrix(0,nrow=2,ncol=20)
mapP[1,1:10] <- 1
mapP[2,11:20] <- 1

## Run the GAPS matrix decomposition
testmap <- gapsMapRun(SimpSim.D, SimpSim.S, FP=mapP,
                      nFactor=3,nEquil = 1000,nSample = 1000)


## Compare fixed patterns to input patterns (after scaling)
summary(t(testmap$Pmean[2:3,] - sweep(mapP,1,apply(mapP,1,sum),FUN="/")))

## End(Not run)
```

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 (A and P) whose product reconstruct the data matrix (D).

Usage

```
gapsRun(D, S, nFactor = "7", simulation_id = "simulation",
  nEquil = "1000", nSample = "1000", nOutR = 1000,
  output_atomic = "FALSE", alphaA = "0.01", nMaxA = "100000",
  max_gibbmass_paraA = "100.0", lambdaA_scale_factor = "1.0",
  alphaP = "0.01", nMaxP = "100000", max_gibbmass_paraP = "100.0",
  lambdaP_scale_factor = "1.0")
```

Arguments

D	data matrix
S	uncertainty matrix (std devs for chi-squared of Log Likelihood)
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)
alphaA	sparsity parameter for A domain
alphaP	sparsity parameter for P domain
max_gibbmass_paraA	limit truncated normal to max size in A
max_gibbmass_paraP	limit truncated normal to max size in P
nMaxA	PRESENTLY UNUSED, future = limit number of atoms in A
nMaxP	PRESENTLY UNUSED, future = limit number of atoms in P
lambdaA_scale_factor	lambda factor in penalized likelihood in A
lambdaP_scale_factor	lambda factor in penalized likelihood in P

Details

The decomposition in GAPS is achieved by finding amplitude and pattern matrices ($\mathbf{A}$ and $\mathbf{P}$, respectively) for which

$$\mathbf{D} = \mathbf{AP} + \Sigma$$

, where Σ is the matrix of uncertainties given by S. The matrices $\mathbf{A}$ and $\mathbf{P}$ are assumed to have the atomic prior described in Sibisi and Skilling (1997) and are found with MCMC sampling.

Value

A list containing:

`Amean`	Sampled mean value of the amplitude matrix $\mathbf{A}$.
`Asd`	Sampled standard deviation of the amplitude matrix $\mathbf{A}$.
`Pmean`	Sampled mean value of the amplitude matrix $\mathbf{P}$.
`Psd`	Sampled standard deviation of the amplitude matrix $\mathbf{P}$.
`atomsAEquil`	Number of atoms in $\mathbf{A}$ during each iteration of the equilibration phase.
`atomsASamp`	Number of atoms in $\mathbf{A}$ during each iteration of the sampling phase.
`atomsPEquil`	Number of atoms in $\mathbf{P}$ during each iteration of the equilibration phase.
`atomsPSamp`	Number of atoms in $\mathbf{P}$ during each iteration of the sampling phase.
`chiSqValues`	Value of chi^2 at each step during equilibration and sampling.
`meanChi2`	Value of chi^2 for Amean and Pmean.

See Also

CoGAPS

Examples

```
## Not run:
## Load data
data(SimpSim)

## Run GAPS matrix decomposition
nIter <- 5000
results <- gapsRun(SimpSim.D, SimpSim.S, nFactor=3,
                   nEquil=nIter, nSample=nIter)

## Plot the results
plotGAPS(results$Amean, results$Pmean, GSFigs)

## End(Not run)
```

GIST.D *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

Ochs, M., Rink, L., Tarn, C., Mburu, S., Taguchi, T., Eisenberg, B., and Godwin, A. (2009). Detection of treatment-induced changes in signaling pathways in gastrointestinal stromal tumors using transcriptomic data. Cancer Res, 69(23), 9125-9132.

GIST.S *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

Ochs, M., Rink, L., Tarn, C., Mburu, S., Taguchi, T., Eisenberg, B., and Godwin, A. (2009). Detection of treatment-induced changes in signaling pathways in gastrointestinal stromal tumors using transcriptomic data. Cancer Res, 69(23), 9125-9132.

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.

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 during the sampling period or equilibration periodfrom one of either A or P as specified in type.

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 teop atome 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 — *Plotter for GAPS decomposition results*

Description

Plots the A and P matrices obtained from the GAPS matrix decomposition.

Usage

```
plotGAPS(A, P, outputPDF="")
```

Arguments

A	The amplitude matrix $\mathbf{A}$ obtained from GAPS.
P	The pattern matrix $\mathbf{P}$ obtained from GAPS.
outputPDF	Name of an pdf file to which the results will be output. (Optional; default="" will output plots to screen).

Note

If the plot option is true in CoGAPS, this function will be called automatically to plot results to the screen.

Author(s)

Elana J. Fertig <efertig@jhmi.edu>

See Also

CoGAPS

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

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

plotSmoothPatterns — *Plot loess smoothed CoGAPS patterns*

Description

Plots the sampled mean value of the pattern matrix $\mathbf{P}$ obtained from the CoGAPS matrix factorization vs. a specificed X value for each sample in the columns of $\mathbf{P}$. Lines plot loess normalized values of $\mathbf{P}$ vs specified X variables.

Usage

```
plotSmoothPatterns(P, x=NULL, breaks=NULL, breakStyle=T, orderP=!all(is.null(x)), plotPTS=F, pointCo
```

Arguments

P	A [p, M] pattern matrix (P.mean) obtained from the CoGAPS matrix factorization.
x	A [M, 1] matrix of values for the X axis for each of the corresponding M columns of P. (Optional: Default: x=1:M)
breaks	A vector of X values at which breaks in plotting should occur. Loess lines fit to data will start and stop at breaks. (Optional: Default: no breaks). May also be specified as an integer to determine the number of equal groups into which to divide the data.
breakStyle	A logical vector. If TRUE, the corresponding break will start a new plot on the row for each pattern. If FALSE, a vertical line will demarcate the break point. (Optional: Defaults to all hard breaks). Note, if one logical value is used, that value will determine the break type at each break point.

orderP	A logical value. If TRUE, vertical ordering of patterns will be determined in order of the value of x at which they peak. If FALSE, vertical ordering will be determined by the rows in the P matrix. (Optional: Default: FALSE)
plotPTS	A logical value. If TRUE, plot will include points for each value of the P matrix in addition to the loess smoothed curve. If FALSE, only the loess smoothed values of P will be plotted. (Optional: Default: FALSE)
pointCol	Color of points of the P matrix plotted when plotPTS=TRUE. (Optional: Default: black)
lineCol	Color of loess smoothed values of the P matrix. (Optional: Default: grey)
add	A logical value. If TRUE, plot will be added to existing graphics device. If FALSE, will create a new graphics device. (Optional: Default: FALSE)
...	Additional arguments to plotting functions.

Author(s)

Genevieve Stein-O'Brien <gsteino1@jhmi.edu>

See Also

CoGAPS

Examples

```
## Not run:
# create simulated data
P <- rbind(1:10 + rnorm(10), seq(from=10,to=1) + rnorm(10))

# saved as PDF since figure margins are often too large for the null device with this function
# and the null device may also have trouble with the overlay
pdf(Test.pdf, width=10)
plotSmoothPatterns(P=P, x=rep(seq(from=1,to=10,by=2),each=2), breaks=3, breakStyle=c(F,T,T), plotPTS=T)

# demonstrating the overlay of the plot
plotSmoothPatterns(P=P, x=rep(seq(from=1,to=10,by=2),each=2), breaks=c(0.992,3.660,6.340,9.010),  breakStyle=c(
dev.off()

## End(Not run)
```

---

reorderByPatternMatch *Match two sets of patterns found with CoGAPS*

Description

Matches two sets of pattern matrices (of the same size) found with CoGAPS. Matches are identified by finding identifying subsequently decreasing correlations between patterns in the respective matrices.

Usage

```
reorderByPatternMatch(P, matchTo)
```

Arguments

P	Pattern matrix for which rows will be arranged to match the matrix in matchTo
matchTo	Pattern matrix to which P is matched.

Value

Pattern matrix derived from reordering columns of P

---

residuals *residuals calculate residuals and produce heatmap*

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 *Simulated data*

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 *Simulated data*

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 *Simulated data*

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 *Simulated data*

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 *Gene sets defined by transcription factors defined from TRANSFAC.*

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

Ochs, M., Rink, L., Tarn, C., Mburu, S., Taguchi, T., Eisenberg, B., and Godwin, A. (2009). Detection of treatment-induced changes in signaling pathways in gastrointestinal stromal tumors using transcriptomic data. Cancer Res, 69(23), 9125-9132.

Index