Package ‘CoGAPS’

April 1, 2024

Version 3.22.0
Date 2023-03-18
Title Coordinated Gene Activity in Pattern Sets
Author Jeanette Johnson, Ashley Tsang, Jacob Mitchell, Thomas Sherman, Wai-shing Lee, Conor Kelton, Ondrej Maxian, Jacob Carey, Genevieve Stein-O’Brien, Michael Considine, Maggie Wodicka, 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>,
Thomas D. Sherman <tomsherman159@gmail.com>,
Jeanette Johnson <jjohn450@jhmi.edu>,
Dmitrijs Lvovs <dlvovs1@jh.edu>

Depends R (>= 3.5.0)
Imports BiocParallel, cluster, methods, gplots, graphics, grDevices,
 RColorBrewer, Rcpp, S4Vectors, SingleCellExperiment, stats,
 SummarizedExperiment, tools, utils, rhdf5, biomaRt, dplyr,
 fgsea, forcats, ggplot2, msigdb
Suggests testthat, knitr, rmarkdown, BiocStyle, SeuratObject
LinkingTo Rcpp, BH
VignetteBuilder knitr
LazyLoad true
License BSD_3_clause + file LICENSE
biocViews GeneExpression, Transcription, GeneSetEnrichment,
  DifferentialExpression, Bayesian, Clustering, TimeCourse,
  RNASEq, Microarray, MultipleComparison, DimensionReduction,
  ImmunoOncology
NeedsCompilation yes
RoxygenNote 7.2.3
Encoding UTF-8
Collate 'class-CogapsParams.R' 'CoGAPS.R' 'DistributedCogaps.R'
  'HelperFunctions.R' 'Package.R' 'RcppExports.R' 'SubsetData.R'
  'class-CogapsResult.R' 'data.R' 'methods-CogapsParams.R'
  'methods-CogapsResult.R'
git_url https://git.bioconductor.org/packages/CoGAPS
git_branch RELEASE_3_18
git_last_commit 4118fd6
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-04-01

R topics documented:

CoGAPS-package ........................................ 4
binaryA .................................................. 4
buildReport ........................................... 5
calcCoGAPSStat ........................................ 5
calcGeneGSStat ......................................... 6
calcZ .................................................. 7
callInternalCoGAPS .................................... 8
checkDataMatrix ....................................... 9
checkInputs ........................................... 9
checkpointsEnabled .................................. 10
CoGAPS ................................................ 10
CogapsParams ......................................... 12
CogapsParams-class ................................... 12
CogapsResult-class ................................... 14
compiledWithOpenMPSupport ......................... 14
computeGeneGSProb ................................ 15
convertDataToMatrix ................................ 16
corcut ................................................. 16
corrToMeanPattern ................................... 17
createCogapsResult .................................. 17
createSets ............................................ 18
distributedCogaps .................................... 18
findConsensusMatrix .................................. 19
fromCSV ............................................... 19
gapsCat ............................................... 20
getAmplitudeMatrix .................................. 20
getClusteredPatterns ................................. 21
getCorrelationToMeanPattern ....................... 21
getDimNames .......................................... 22
getFeatureLoadings .................................. 22
getGeneNames .......................................................... 23
getMeanChiSq .......................................................... 23
getOriginalParameters ............................................... 24
getParam ............................................................... 24
getPatternHallmarks .................................................. 25
getPatternMatrix ....................................................... 25
getRetinaSubset ....................................................... 26
getSampleFactors ..................................................... 27
getSampleNames ....................................................... 27
getSubsets ............................................................ 28
getUnmatchedPatterns ............................................... 28
getValueOrRds .......................................................... 29
getVersion .............................................................. 29
GIST.data_frame ....................................................... 30
GIST.matrix ............................................................ 30
GIST.result ............................................................. 30
GIST.uncertainty ...................................................... 30
GWCoGAPS .............................................................. 31
initialize,CogapsParams-method .................................... 32
initialize,CogapsResult-method ..................................... 33
isRdsFile ............................................................... 34
MANOVA ................................................................. 34
modsimdata ............................................................. 35
modsimresult .......................................................... 35
ncolHelper ............................................................. 35
nrowHelper ............................................................. 36
parseExtraParams ...................................................... 36
patternMarkers .......................................................... 37
patternMatch ............................................................ 37
plotPatternHallmarks .................................................. 38
plotPatternMarkers .................................................... 38
plotResiduals ........................................................... 39
reconstructGene .......................................................... 40
sampleUniformly .......................................................... 41
sampleWithAnnotationWeights ....................................... 41
sampleWithExplicitSets .............................................. 42
scCoGAPS ............................................................... 42
setAnnotationWeights .................................................. 44
setDistributedParams ................................................... 44
setFixedPatterns .......................................................... 45
setParam ............................................................... 46
startupMessage .......................................................... 47
stitchTogether .......................................................... 47
supported ............................................................... 48
toCSV ................................................................. 48

Index ................................................................. 49
binaryA

binary heatmap for standardized feature matrix

Description

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(object, threshold = 3)

## S4 method for signature 'CogapsResult'
binaryA(object, threshold = 3)

Arguments

object an object of type CogapsResult

threshold the number of standard deviations above zero that an element of Amean must be to get a value of 1

---

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.

Package: CoGAPS  
Type: Package  
Version: 2.99.0  
Date: 2018-01-24  
License: LGPL

Author(s)

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

References

**buildReport**

**Value**
plots a heatmap of the A Matrix

**Examples**
```r
data(GIST)
# too expensive to call since it plots
# binaryA(GIST.result, threshold=3)
```

---

**buildReport**

**Information About Package Compilation**

**Description**
Information About Package Compilation

**Usage**
```r
buildReport()
```

**Details**
returns information about how the package was compiled, i.e. which compiler/version was used, which compile time options were enabled, etc...

**Value**
string containing build report

**Examples**
```r
CoGAPS::buildReport()
```

---

**calcCoGAPSStat**

*calculate statistic on sets of measurements (genes) or samples*

**Description**
calculates a statistic to determine if a pattern is enriched in a particular set of measurements or samples.
Usage

calcCoGAPSStat(
    object,
    sets = NULL,
    whichMatrix = "featureLoadings",
    numPerm = 1000,
    ...
)

## S4 method for signature 'CogapsResult'
calcCoGAPSStat(
    object,
    sets = NULL,
    whichMatrix = "featureLoadings",
    numPerm = 1000,
    ...
)

Arguments

object an object of type CogapsResult
sets list of sets of measurements/samples
whichMatrix either "featureLoadings" or "sampleFactors" indicating which matrix to calculate the statistics for
numPerm number of permutations to use when calculating p-value
... handles old arguments for backwards compatibility

Value
gene set statistics for each column of A

Description
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(
    object,
    GStoGenes,
    numPerm,
    Pw = rep(1, ncol(object@featureLoadings)),
    ...
calcZ

nullGenes = FALSE

## S4 method for signature 'CogapsResult'
calcGeneGSStat(
  object,
  GStoGenes,
  numPerm,
  Pw = rep(1, ncol(object@featureLoadings)),
  nullGenes = FALSE
)

Arguments

object an object of type CogapsResult
GStoGenes data.frame or list with gene sets
numPerm number of permutations for null
Pw weight on genes
nullGenes logical indicating gene adjustment

Value
gene similarity statistic

calcZ compute z-score matrix

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

Usage
calcZ(object, whichMatrix)

## S4 method for signature 'CogapsResult'
calcZ(object, whichMatrix)

Arguments

object an object of type CogapsResult
whichMatrix either "featureLoadings" or "sampleFactors" indicating which matrix to calculate the z-score for

Value

matrix of z-scores
### Examples

```r
data(GIST)
featureZScore <- calcZ(GIST.result, "featureLoadings")
```

### Description

Make correct call to internal CoGAPS dispatch function. CoGAPS could be called directly, but to avoid any re-entrant behavior this function is called instead. It is a light wrapper around cogaps_cpp that handles setting the distributed parameters.

### Usage

```r
callInternalCoGAPS(data, allParams, uncertainty, subsetIndices, workerID)
```

### Arguments

- **data**: data in a supported format
- **allParams**: list of all parameters
- **uncertainty**: uncertainty of data in the same format
- **index**: index for which subset to run on
- **sets**: list of all subsets
- **geneNames**: names of all genes
- **sampleNames**: names of all samples
- **fixedMatrix**: matrix of matched patterns

### Value

CogapsResult object
**checkDataMatrix**

check that provided data is valid

**Description**

check that provided data is valid

**Usage**

`checkDataMatrix(data, uncertainty, params)`

**Arguments**

- **data**: data matrix
- **uncertainty**: uncertainty matrix, can be null
- **params**: CogapsParams object

**Value**

throws an error if data has problems

**checkInputs**

check that all inputs are valid

**Description**

check that all inputs are valid

**Usage**

`checkInputs(data, uncertainty, allParams)`

**Arguments**

- **data**: data matrix
- **uncertainty**: uncertainty matrix, can be null
- **allParams**: list of all parameters

**Value**

throws an error if inputs are invalid
checkpointsEnabled  
Check if package was built with checkpoints enabled

**Description**
Check if package was built with checkpoints enabled

**Usage**
checkpointsEnabled()

**Value**
true/false if checkpoints are enabled

**Examples**
CoGAPS::checkpointsEnabled()

---

CoGAPS  
*CoGAPS Matrix Factorization Algorithm*

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

**Usage**
CoGAPS(
    data,
    params = new("CogapsParams"),
    nThreads = 1,
    messages = TRUE,
    outputFrequency = 1000,
    uncertainty = NULL,
    checkpointOutFile = "gaps_checkpoint.out",
    checkpointInterval = 0,
    checkpointInFile = NULL,
    transposeData = FALSE,
    BPPARAM = NULL,
    workerID = 1,
    asynchronousUpdates = TRUE,
    nSnapshots = 0,
    snapshotPhase = "sampling",
    ...
)
Arguments

- **data**: File name or R object (see details for supported types)
- **params**: CogapsParams object
- **nThreads**: maximum number of threads to run on
- **messages**: T/F for displaying output
- **outputFrequency**: number of iterations between each output (set to 0 to disable status updates, other output is controlled by `messages`)
- **uncertainty**: uncertainty matrix - either a matrix or a supported file type
- **checkpointOutFile**: name of the checkpoint file to create
- **checkpointInterval**: number of iterations between each checkpoint (set to 0 to disable checkpoints)
- **checkpointInFile**: if this is provided, CoGAPS runs from the checkpoint contained in this file
- **transposeData**: T/F for transposing data while reading it in - useful for data that is stored as samples x genes since CoGAPS requires data to be genes x samples
- **BPPARAM**: BiocParallel backend
- **workerID**: if calling CoGAPS in parallel the worker ID can be specified, only worker 1 prints output and each worker outputs when it finishes, this is not necessary when using the default parallel methods (i.e. distributed CoGAPS) but only when the user is manually calling CoGAPS in parallel
- **asynchronousUpdates**: enable asynchronous updating which allows for multi-threaded runs
- **nSnapshots**: how many snapshots to take in each phase, setting this to 0 disables snapshots
- **snapshotPhase**: which phase to take snapshots in e.g. "equilibration", "sampling", "all"
- **...**: allows for overwriting parameters in params

Details

The supported R types are: matrix, data.frame, SummarizedExperiment, SingleCellExperiment. The supported file types are csv, tsv, and mtx.

Value

CogapsResult object

Examples

```r
# Running from R object
data(GIST)
resultA <- CoGAPS(GIST.data_frame, nIterations=25)

# Running from file name
gist_path <- system.file("extdata/GIST.mtx", package="CoGAPS")
```
resultB <- CoGAPS(gist_path, nIterations=25)

# Setting Parameters
params <- new("CogapsParams")
params <- setParam(params, "nPatterns", 3)
resultC <- CoGAPS(GIST.data_frame, params, nIterations=25)

---

**CogapsParams**  
*CogapsParams constructor*

**Description**

create a CogapsParams object

**Usage**

*CogapsParams(...)*

**Arguments**

...  
parameters for the initialization method

**Value**

CogapsParams object

**Examples**

params <- CogapsParams(nPatterns=10)
params

---

**CogapsParams-class**  
*CogapsParams*

**Description**

Encapsulates all parameters for the CoGAPS algorithm
Slots

nPatterns number of patterns CoGAPS will learn
nIterations number of iterations for each phase of the algorithm
alphaA sparsity parameter for feature matrix
alphaP sparsity parameter for sample matrix
maxGibbsMassA atomic mass restriction for feature matrix
maxGibbsMassP atomic mass restriction for sample matrix
seed random number generator seed
sparseOptimization speeds up performance with sparse data (roughly >80 default uncertainty
distributed either "genome-wide" or "single-cell" indicating which distributed algorithm should
be used
nSets [distributed parameter] number of sets to break data into
cut [distributed parameter] number of branches at which to cut dendrogram used in pattern matching
minNS [distributed parameter] minimum of individual set contributions a cluster must contain
maxNS [distributed parameter] maximum of individual set contributions a cluster can contain
explicitSets [distributed parameter] specify subsets by index or name
samplingAnnotation [distributed parameter] specify categories along the rows (cols) to use for weighted sampling
samplingWeight [distributed parameter] weights associated with samplingAnnotation
subsetIndices set of indices to use from the data
subsetDim which dimension (1=rows, 2=cols) to subset
geneNames vector of names of genes in data
sampleNames vector of names of samples in data
fixedPatterns fix either 'A' or 'P' matrix to these values, in the context of distributed CoGAPS (GWCoGAPS/scCoGAPS), the first phase is skipped and fixedPatterns is used for all sets - allowing manual pattern matching, as well as fixed runs of standard CoGAPS
whichMatrixFixed either 'A' or 'P', indicating which matrix is fixed
takePumpSamples whether or not to take PUMP samples
checkpointInterval how many iterations between each checkpoint (set to 0 to disable)
checkpointInFile file path to load checkpoint from
checkpointOutFile file path where checkpoint should be written to
CogapsResult-class  

**CogapsResult**

**Description**
Contains all output from Cogaps run

**Slots**
- factorStdDev  std dev of the sampled P matrices
- loadingStdDev  std dev of the sampled A matrices

---

compiledWithOpenMPSupport

*Check if compiler supported OpenMP*

**Description**
Check if compiler supported OpenMP

**Usage**
compiledWithOpenMPSupport()

**Value**
true/false if OpenMP was supported

**Examples**
CoGAPS::compiledWithOpenMPSupport()
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.

Usage

computeGeneGSProb(
  object,
  GStoGenes,
  numPerm = 500,
  Pw = rep(1, ncol(object@featureLoadings)),
  PwNull = FALSE
)

## S4 method for signature 'CogapsResult'
computeGeneGSProb(
  object,
  GStoGenes,
  numPerm = 500,
  Pw = rep(1, ncol(object@featureLoadings)),
  PwNull = FALSE
)

Arguments

- **object**: an object of type CogapsResult
- **GStoGenes**: data.frame or list with gene sets
- **numPerm**: number of permutations for null
- **Pw**: weight on genes
- **PwNull**: logical indicating gene adjustment

Value

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

convert any acceptable data input to a numeric matrix

Description
convert supported R objects containing the data to a numeric matrix, if data is a file name do nothing.
Exits with an error if data is not a supported type.

Usage
convertDataToMatrix(data)

Arguments
data  data input

Value
data matrix

corcut  

cluster patterns together

Description
cluster patterns together

Usage
corcut(allPatterns, cut, minNS)

Arguments
allPatterns  matrix of all patterns across subsets
cut  number of branches at which to cut dendrogram
minNS  minimum of individual set contributions a cluster must contain

Value
patterns listed by which cluster they belong to
**corrToMeanPattern**

*calculate correlation of each pattern in a cluster to the cluster mean*

**Description**

calculate correlation of each pattern in a cluster to the cluster mean

**Usage**

`corrToMeanPattern(cluster)`

**Value**

correlation of each pattern

---

**createCogapsResult**

*convert list output from c++ code to a CogapsResult object*

**Description**

convert list output from c++ code to a CogapsResult object

**Usage**

`createCogapsResult(returnList, allParams)`

**Arguments**

- `returnList`: list from cogaps_cpp
- `allParams`: list of all parameters

**Value**

CogapsResult object
createSets  
partition genes/samples into subsets

Description

either genes or samples or partitioned depending on the type of distributed CoGAPS (i.e. genome-wide or single-cell)

Usage

createSets(data, allParams)

Arguments

data either file name or matrix
allParams list of all CoGAPS parameters

Value

list of sorted subsets of either genes or samples

distributedCogaps  
CoGAPS Distributed Matrix Factorization Algorithm

Description

runs CoGAPS over subsets of the data and stitches the results back together

Usage

distributedCogaps(data, allParams, uncertainty)

Arguments

data File name or R object (see details for supported types)
allParams list of all parameters used in computation
uncertainty uncertainty matrix (same supported types as data)

Details

For file types CoGAPS supports csv, tsv, and mtx

Value

list
findConsensusMatrix

Description
find the consensus pattern matrix across all subsets

Usage
findConsensusMatrix(unmatchedPatterns, gapsParams)

Arguments
unmatchedPatterns
list of all unmatched pattern matrices from initial run of CoGAPS

гapsParams
list of all CoGAPS parameters

Value
matrix of consensus patterns

fromCSV

Description
read CoGAPS Result object from a directory with a set of csvs see toCSV

Usage
fromCSV(save_location = ".")

Arguments
save_location
directory to read from

Value
CogapsResult object
getAmplitudeMatrix

Description
return Amplitude matrix from CogapsResult object

Usage
getAmplitudeMatrix(object)

## S4 method for signature 'CogapsResult'
getAmplitudeMatrix(object)

Arguments
object an object of type CogapsResult

Value
amplitude matrix

Examples

data(GIST)
amplitudeMatrix <- getAmplitudeMatrix(GIST.result)

---

gapsCat wrapper around cat

Description
cleans up message printing

Usage
gapsCat(allParams, ...)

Arguments
allParams all cogaps parameters
... arguments forwarded to cat

Value
conditionally print message

---

gapGetAmplitudeMatrix
**getClusteredPatterns**

*return clustered patterns from set of all patterns across all subsets*

**Description**

return clustered patterns from set of all patterns across all subsets

**Usage**

```r
getClusteredPatterns(object)
```

**Arguments**

- `object` an object of type CogapsResult

**Value**

CogapsParams object

**Examples**

```r
data(GIST)
clusteredPatterns <- getClusteredPatterns(GIST.result)
```

**getCorrelationToMeanPattern**

*return correlation between each pattern and the cluster mean*

**Description**

return correlation between each pattern and the cluster mean

**Usage**

```r
getCorrelationToMeanPattern(object)
```

**Arguments**

- `object` an object of type CogapsResult
getFeatureLoadings

Value
CogapsParams object

Examples
data(GIST)
corrToMeanPattern <- getCorrelationToMeanPattern(GIST.result)

getDimNames extracts gene/sample names from the data

Description
extracts gene/sample names from the data

Usage
getDimNames(data, allParams)

Arguments
data data matrix
allParams list of all parameters

Value
list of all parameters with added gene names

getFeatureLoadings return featureLoadings matrix from CogapsResult object

Description
return featureLoadings matrix from CogapsResult object

Usage
getFeatureLoadings(object)

## S4 method for signature 'CogapsResult'
getFeatureLoadings(object)

Arguments
object an object of type CogapsResult
**getGeneNames**

**Value**

featureLoadings matrix

**Examples**

data(GIST)
flLoadings <- getFeatureLoadings(GIST.result)

geneNames <- getGeneNames(flLoadings)

**Description**

extract gene names from data

**Usage**

geneNames(data, transpose)

**Value**

vector of gene names

---

**getMeanChiSq**

return chi-sq of final matrices

**Description**

return chi-sq of final matrices

**Usage**

getMeanChiSq(object)

## S4 method for signature 'CogapsResult'
getMeanChiSq(object)

**Arguments**

object an object of type CogapsResult

**Value**

chi-sq error

**Examples**

data(GIST)
getMeanChiSq(GIST.result)
getOriginalParameters  return original parameters used to generate this result

Description

return original parameters used to generate this result

Usage

getOriginalParameters(object)

## S4 method for signature 'CogapsResult'
getOriginalParameters(object)

Arguments

object an object of type CogapsResult

Value

CogapsParams object

Examples

data(GIST)
params <- getOriginalParameters(GIST.result)

getParam  get the value of a parameter

Description

get the value of a parameter

Usage

getParam(object, whichParam)

## S4 method for signature 'CogapsParams'
getParam(object, whichParam)

Arguments

object an object of type CogapsParams
whichParam a string with the name of the requested parameter
**getPatternHallmarks**

**Value**

the value of the parameter

**Examples**

```r
params <- new("CogapsParams")
getParam(params, "seed")
```

---

**getPatternHallmarks**

generate statistics associating patterns with MSigDB hallmark gene sets

**Description**

generate statistics associating patterns with MSigDB hallmark gene sets

**Usage**

```r
getPatternHallmarks(object)
```

## S4 method for signature 'CogapsResult'

getPatternHallmarks(object)

**Arguments**

object an object of type CogapsResult

**Value**

dataframe of hallmark info

---

**getPatternMatrix**

return pattern matrix from CogapsResult object

**Description**

return pattern matrix from CogapsResult object

**Usage**

```r
getPatternMatrix(object)
```

## S4 method for signature 'CogapsResult'

generate statistics associating patterns with MSigDB hallmark gene sets

getPatternMatrix(object)
getRetinaSubset

Arguments

object an object of type CogapsResult

Value

pattern matrix

Examples

data(GIST)
patternMatrix <- getPatternMatrix(GIST.result)

getRetinaSubset get specified number of retina subsets

Description

combines retina subsets from extdata directory

Usage

getRetinaSubset(n = 1)

Arguments

n number of subsets to use

Value

matrix of RNA counts

Examples

retSubset <- getRetinaSubset()
dim(retSubset)
**getSampleFactors**

*Description*

return sampleFactors matrix from CogapsResult object

*Usage*

```
getSampleFactors(object)
```

## S4 method for signature 'CogapsResult'

getSampleFactors(object)

*Arguments*

- `object` an object of type CogapsResult

*Value*

sampleFactors matrix

*Examples*

```r
data(GIST)
sFactors <- getSampleFactors(GIST.result)
```

**getSampleNames**

*Description*

extract sample names from data

*Usage*

```
getSampleNames(data, transpose)
```

*Value*

vector of sample names
getSubsets  

*return the names of the genes (samples) in each subset*

**Description**

return the names of the genes (samples) in each subset

**Usage**

getSubsets(object)

```
## S4 method for signature 'CogapsResult'
getSubsets(object)
```

**Arguments**

- `object` an object of type CogapsResult

**Value**

CogapsParams object

**Examples**

```r
data(GIST)
subsets <- getSubsets(GIST.result)
```

getUnmatchedPatterns  

*return unmatched patterns from each subset*

**Description**

return unmatched patterns from each subset

**Usage**

getUnmatchedPatterns(object)

```
## S4 method for signature 'CogapsResult'
getUnmatchedPatterns(object)
```

**Arguments**

- `object` an object of type CogapsResult
getValueOrRds

Value
CogapsParams object

Examples
data(GIST)
unmatchedPatterns <- getUnmatchedPatterns(GIST.result)

getNextVersion

give input that might be an RDS file

Description
give input that might be an RDS file

Usage
gevalValueOrRds(input)

Arguments
input some user input

Value
if input is an RDS file, read it - otherwise return input

getNextVersion
return version number used to generate this result

Description
return version number used to generate this result

Usage
gevalVersion(object)

Arguments
object an object of type CogapsResult
GIST.uncertainty

Value
version number

Examples
data(GIST)
getVersion(GIST.result)

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

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

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

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

GIST.result
CoGAPS result from running on GIST dataset

Description
CoGAPS result from running on GIST dataset

GIST.uncertainty
GIST gene expression uncertainty matrix from Ochs et al. (2009)

Description
GIST gene expression uncertainty matrix from Ochs et al. (2009)
**Description**

wrapper around genome-wide distributed algorithm for CoGAPS

**Usage**

```
GWCoGAPS(
  data,
  params = new("CogapsParams"),
  nThreads = 1,
  messages = TRUE,
  outputFrequency = 500,
  uncertainty = NULL,
  checkpointOutFile = "gaps_checkpoint.out",
  checkpointInterval = 1000,
  checkpointInFile = NULL,
  transposeData = FALSE,
  BPPARAM = NULL,
  workerID = 1,
  asynchronousUpdates = FALSE,
  ...
)
```

**Arguments**

- **data**: File name or R object (see details for supported types)
- **params**: CogapsParams object
- **nThreads**: maximum number of threads to run on
- **messages**: T/F for displaying output
- **outputFrequency**: number of iterations between each output (set to 0 to disable status updates, other output is controlled by @code messages)
- **uncertainty**: uncertainty matrix - either a matrix or a supported file type
- **checkpointOutFile**: name of the checkpoint file to create
- **checkpointInterval**: number of iterations between each checkpoint (set to 0 to disable checkpoints)
- **checkpointInFile**: if this is provided, CoGAPS runs from the checkpoint contained in this file
- **transposeData**: T/F for transposing data while reading it in - useful for data that is stored as samples x genes since CoGAPS requires data to be genes x samples
- **BPPARAM**: BiocParallel backend
workerID
if calling CoGAPS in parallel the worker ID can be specified, only worker 1
prints output and each worker outputs when it finishes, this is not neccessary
when using the default parallel methods (i.e. distributed CoGAPS) but only
when the user is manually calling CoGAPS in parallel

asynchronousUpdates
enable asynchronous updating which allows for multi-threaded runs

... allows for overwriting parameters in params

Value
CogapsResult object

Examples
## Not run:
data(GIST)
params <- new("CogapsParams")
params <- setDistributedParams(params, nSets=2)
params <- setParam(params, "nIterations", 100)
params <- setParam(params, "nPatterns", 3)
result <- GWCoGAPS(GIST.matrix, params, BPPARAM=BiocParallel::SerialParam())

## End(Not run)
Description

Constructor for CogapsResult

Usage

## S4 method for signature 'CogapsResult'
initialize(
  .Object,
  Amean,
  Pmean,
  Asd,
  Psd,
  meanChiSq,
  geneNames,
  sampleNames,
  diagnostics = NULL,
  ...
)

Arguments

/Object CogapsResult object
Amean mean of sampled A matrices
Pmean mean of sampled P matrices
Asd std dev of sampled A matrices
Psd std dev of sampled P matrices
meanChiSq mean value of ChiSq statistic
geneNames names of genes in data
sampleNames names of samples in data
diagnostics assorted diagnostic reports from the run
... initial values for slots

Value

initialized CogapsResult object
### isRdsFile

**checks if file is rds format**

**Description**

checks if file is rds format

**Usage**

```r
isRdsFile(file)
```

**Arguments**

- `file` path to file

**Value**

TRUE if file is .rds, FALSE if not

---

### MANOVA

**MANOVA statistical test for patterns between sample groups**

**Description**

MANOVA statistical test–wraps base R manova

**Usage**

```r
MANOVA(interestedVariables, object)
```

**Arguments**

- `interestedVariables` study design for manova
- `object` CogapsResult object

**Value**

list of manova fit results
**modsimdata**

**modsimdata**  
*Toy example to run CoGAPS on.*

**Description**
- V1..V20. some variables, for example levels of gene expression

**Usage**
```
data(modsimdata)
```

**Format**
- 'data.frame': 25 obs. of 20 variables.

---

**modsimresult**  
*Result of applying CoGAPS on the Toy example.*

**Description**
- Result of applying CoGAPS on the Toy example.

**Usage**
```
data(modsimresult)
```

**Format**
- S4 class ‘CogapsResult’ [package “CoGAPS”] with 7 slots.

---

**ncolHelper**  
*get number of columns from supported file name or matrix*

**Description**
- get number of columns from supported file name or matrix

**Usage**
```
ncolHelper(data)
```

**Arguments**
- data: either a file name or a matrix

**Value**
- number of columns
**nrowHelper**

*get number of rows from supported file name or matrix*

**Description**

get number of rows from supported file name or matrix

**Usage**

```r
nrowHelper(data)
```

**Arguments**

- `data` either a file name or a matrix

**Value**

number of rows

---

**parseExtraParams**

*parse parameters passed through the ... variable*

**Description**

parse parameters passed through the ... variable

**Usage**

```r
parseExtraParams(allParams, extraParams)
```

**Arguments**

- `allParams` list of all parameters
- `extraParams` list of parameters in ...

**Value**

allParams with any valid parameters in extraParams added

**Note**

will halt with an error if any parameters in extraParams are invalid
patternMarkers

compute pattern markers statistic

Description

calculate the most associated pattern for each gene

Usage

patternMarkers(object, threshold = "all", lp = NA, axis = 1)

## S4 method for signature 'CogapsResult'

patternMarkers(object, threshold = "all", lp = NA, axis = 1)

Arguments

object an object of type CogapsResult

threshold the type of threshold to be used. The default "all" will distribute genes into pattern with the lowest ranking. The "cut" thresholds 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.

axis either 1 or 2, specifying if pattern markers should be calculated using the rows of the data (1) or the columns of the data (2)

Value

By default a non-overlapping list of genes associated with each lp.

Examples

data(GIST)

pm <- patternMarkers(GIST.result)

patternMatch

Match Patterns Across Multiple Runs

Description

Match Patterns Across Multiple Runs

Usage

patternMatch(allPatterns, gapsParams)
plotPatternMarkers

Arguments
- **allPatterns**: matrix of patterns stored in the columns
- **gapsParams**: CoGAPS parameters object

Value
- a matrix of consensus patterns

plotPatternHallmarks *generate a barchart of most significant hallmark sets for a pattern*

Description
- generate a barchart of most significant hallmark sets for a pattern

Usage
- `plotPatternHallmarks(object, patternhallmarks, whichpattern = 1)`

## S4 method for signature 'CogapsResult,list,numeric'
- `plotPatternHallmarks(object, patternhallmarks, whichpattern = 1)`

Arguments
- **object**: an object of type CogapsResult
- **patternhallmarks**: output from getPatternHallmarks
- **whichpattern**: which pattern to generate bar chart for

Value
- image object of barchart

plotPatternMarkers *heatmap of original data clustered by pattern markers statistic*

Description
- heatmap of original data clustered by pattern markers statistic
Usage

plotPatternMarkers(
  object,
  data,
  patternMarkers,
  patternPalette,
  sampleNames,
  samplePalette = NULL,
  heatmapCol = bluered,
  colDendrogram = TRUE,
  scale = "row",
  ...
)

Arguments

object an object of type CogapsResult
data the original data as a matrix
patternMarkers pattern markers to be plotted, as generated by 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
heatmapCol palette giving color scheme for heatmap
colDendrogram logical indicating whether to display sample dendrogram
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

Description

calculate residuals and produce heatmap
Usage

plotResiduals(object, data, uncertainty = NULL)

## S4 method for signature 'CogapsResult'
plotResiduals(object, data, uncertainty = NULL)

Arguments

object an object of type CogapsResult
data original data matrix run through GAPS
uncertainty original standard deviation matrix run through GAPS

Value

creates a residual plot

Examples

data(GIST)
# too expensive to call since it plots
# plotResiduals(GIST.result, GIST.matrix)

reconstructGene

Description

reconstruct gene

Usage

reconstructGene(object, genes = NULL)

## S4 method for signature 'CogapsResult'
reconstructGene(object, genes = NULL)

Arguments

object an object of type CogapsResult
genpes an index of the gene or genes of interest

Value

the D' estimate of a gene or set of genes

Examples

data(GIST)
estimatedD <- reconstructGene(GIST.result)
**sampleUniformly**
subset data by uniformly partitioning rows (cols)

**Description**
subset data by uniformly partitioning rows (cols)

**Usage**
sampleUniformly(allParams, total, setSize)

**Arguments**
- allParams: list of all CoGAPS parameters
- total: total number of rows (cols) that are being partitioned
- setSize: the size of each subset of the total

**Value**
list of subsets

**sampleWithAnnotationWeights**
subset rows (cols) proportional to the user provided weights

**Description**
subset rows (cols) proportional to the user provided weights

**Usage**
sampleWithAnnotationWeights(allParams, setSize)

**Arguments**
- allParams: list of all CoGAPS parameters
- setSize: the size of each subset of the total

**Value**
list of subsets
sampleWithExplicitSets  *use user provided subsets*

**Description**

use user provided subsets

**Usage**

```r
sampleWithExplicitSets(allParams)
```

**Arguments**

- `allParams`: list of all CoGAPS parameters
- `total`: total number of rows (cols) that are being partitioned

**Value**

list of subsets

---

scCoGAPS  *Single Cell CoGAPS*

**Description**

wrapper around single-cell distributed algorithm for CoGAPS

**Usage**

```r
scCoGAPS(
  data,
  params = new("CogapsParams"),
  nThreads = 1,
  messages = TRUE,
  outputFrequency = 500,
  uncertainty = NULL,
  checkpointOutFile = "gaps_checkpoint.out",
  checkpointInterval = 1000,
  checkpointInFile = NULL,
  transposeData = FALSE,
  BPPARAM = NULL,
  workerID = 1,
  asynchronousUpdates = FALSE,
  ...
)
```
Arguments

- **data**: File name or R object (see details for supported types)
- **params**: CogapsParams object
- **nThreads**: maximum number of threads to run on
- **messages**: T/F for displaying output
- **outputFrequency**: number of iterations between each output (set to 0 to disable status updates, other output is controlled by @code messages)
- **uncertainty**: uncertainty matrix - either a matrix or a supported file type
- **checkpointOutFile**: name of the checkpoint file to create
- **checkpointInterval**: number of iterations between each checkpoint (set to 0 to disable checkpoints)
- **checkpointInFile**: if this is provided, CoGAPS runs from the checkpoint contained in this file
- **transposeData**: T/F for transposing data while reading it in - useful for data that is stored as samples x genes since CoGAPS requires data to be genes x samples
- **BPARAM**: BioParallel backend
- **workerID**: if calling CoGAPS in parallel the worker ID can be specified, only worker 1 prints output and each worker outputs when it finishes, this is not necessary when using the default parallel methods (i.e. distributed CoGAPS) but only when the user is manually calling CoGAPS in parallel
- **asynchronousUpdates**: enable asynchronous updating which allows for multi-threaded runs
- ... allows for overwriting parameters in params

Value

CogapsResult object

Examples

```r
## Not run:
data(GIST)
params <- new("CogapsParams")
params <- setDistributedParams(params, nSets=2)
params <- setParam(params, "nIterations", 100)
params <- setParam(params, "nPatterns", 3)
result <- scCoGAPS(t(GIST.matrix), params, BPARAM=BiocParallel::SerialParam())
```

## End(Not run)
setAnnotationWeights  set the annotation labels and weights for subsetting the data

Description
these parameters are interrelated so they must be set together

Usage
setAnnotationWeights(object, annotation, weights)

## S4 method for signature 'CogapsParams'
setAnnotationWeights(object, annotation, weights)

Arguments

object
an object of type CogapsParams

annotation
vector of labels

weights
vector of weights

Value
the modified params object

Examples

params <- new("CogapsParams")
params <- setAnnotationWeights(params, c('a', 'b', 'c'), c(1,2,1))

setDistributedParams  set the value of parameters for distributed CoGAPS

Description
these parameters are interrelated so they must be set together

Usage

setDistributedParams(
  object,
  nSets = NULL,
  cut = NULL,
  minNS = NULL,
  maxNS = NULL
)

setFixedPatterns 45

```r
## S4 method for signature 'CogapsParams'
setDistributedParams(
o, nSets = NULL, cut = NULL, minNS = NULL, maxNS = NULL)
)
```

**Arguments**

- **object**: an object of type `CogapsParams`
- **nSets**: number of sets to break data into
- **cut**: number of branches at which to cut dendrogram used in pattern matching
- **minNS**: minimum of individual set contributions a cluster must contain
- **maxNS**: maximum of individual set contributions a cluster can contain

**Value**

the modified params object

**Examples**

```r
params <- new("CogapsParams")
params <- setDistributedParams(params, 5)
```

---

**Description**

these parameters are interrelated so they must be set together

**Usage**

```r
setFixedPatterns(object, fixedPatterns, whichMatrixFixed)
```

```r
## S4 method for signature 'CogapsParams'
setFixedPatterns(object, fixedPatterns, whichMatrixFixed)
```

**Arguments**

- **object**: an object of type `CogapsParams`
- **fixedPatterns**: values for either the A or P matrix
- **whichMatrixFixed**: either 'A' or 'P' indicating which matrix is fixed
Value

the modified params object

Examples

params <- new("CogapsParams")
data(GIST)
params <- setFixedPatterns(params, getSampleFactors(GIST.result), 'P')

---

setParam set the value of a parameter

Description

set the value of a parameter

Usage

setParam(object, whichParam, value)

## S4 method for signature 'CogapsParams'
setParam(object, whichParam, value)

Arguments

object an object of type CogapsParams
whichParam a string with the name of the parameter to be changed
value the value to set the parameter to

Value

the modified params object

Examples

params <- new("CogapsParams")
params <- setParam(params, "seed", 123)
### startupMessage

**Description**
write start up message

**Usage**
startupMessage(data, allParams)

**Arguments**
- `data`: data set
- `allParams`: list of all parameters

**Value**
message displayed to screen

### stitchTogether

**Description**
concatenate final results across subsets

**Usage**
stitchTogether(result, allParams, sets)

**Arguments**
- `result`: list of CogapsResult object from all runs across subsets
- `allParams`: list of all CoGAPS parameters
- `sets`: indices of sets used to break apart data

**Value**
list with all CoGAPS output
supported  checks if file is supported

Description
checks if file is supported

Usage
supported(file)

Arguments
file  path to file

Value
TRUE if file is supported, FALSE if not

toCSV  save CoGAPS Result object as a set of csvs to directory see fromCSV

Description
save as csv

Usage
toCSV(object, save_location = ".")

## S4 method for signature 'CogapsResult,character'
toCSV(object, save_location = ".")

Arguments
object  CogapsResult object
save_location  directory to write to

Value
none
Index

* datasets
  - modsimdata, 35
  - modsimresult, 35

* internal
  - callInternalCoGAPS, 8
  - checkDataMatrix, 9
  - checkInputs, 9
  - convertDataToMatrix, 16
  - corcut, 16
  - corrToMeanPattern, 17
  - createCogapsResult, 17
  - createSets, 18
  - distributedCogaps, 18
  - gapsCat, 20
  - getDimNames, 22
  - getGeneNames, 23
  - getSampleNames, 27
  - getValueOrRds, 29
  - isRdsFile, 34
  - ncolHelper, 35
  - nrowHelper, 36
  - parseExtraParams, 36
  - patternMatch, 37
  - sampleUniformly, 41
  - sampleWithAnnotationWeights, 41
  - sampleWithExplicitSets, 42
  - startupMessage, 47
  - stitchTogether, 47
  - supported, 48

binaryA, 4
binaryA, CogapsResult-method (binaryA), 4
buildReport, 5

calcCoGAPSStat, 5

calcCoGAPSStat, CogapsResult-method (calcCoGAPSStat), 5

calcGeneGSStat, 6

calcGeneGSStat, CogapsResult-method (calcGeneGSStat), 6

calcZ, 7
calcZ, CogapsResult-method (calcZ), 7
callInternalCoGAPS, 8
checkDataMatrix, 9
checkInputs, 9
checkpointsEnabled, 10
CoGAPS, 10
CoGAPS-package, 4
CogapsParams, 12
CogapsParams-class, 12
CogapsResult-class, 14
compiledWithOpenMPSupport, 14
computeGeneGSProb, 15
computeGeneGSProb, CogapsResult-method (computeGeneGSProb), 15
convertDataToMatrix, 16
corcut, 16
corrToMeanPattern, 17
createCogapsResult, 17
createSets, 18

distributedCogaps, 18

findConsensusMatrix, 19
fromCSV, 19
fromCSV, character-method (fromCSV), 19
gapsCat, 20
getAmplitudeMatrix, 20
getAmplitudeMatrix, CogapsResult-method (getAmplitudeMatrix), 20
getClusteredPatterns, 21
getClusteredPatterns, CogapsResult-method (getClusteredPatterns), 21

getCorrelationToMeanPattern, 21
getCorrelationToMeanPattern, CogapsResult-method (getCorrelationToMeanPattern), 21

getDimNames, 22
getFeatureLoadings, 22
getIndex

getFeatureLoadings,CogapsResult-method (getFeatureLoadings), 22
getGeneNames, 23
getMeanChiSq, 23
getMeanChiSq,CogapsResult-method (getMeanChiSq), 23
getOriginalParameters, 24
getOriginalParameters,CogapsResult-method (getOriginalParameters), 24
getParam, 24
getParam,CogapsParams-method (getParam), 24
getPatternHallmarks, 25
getPatternHallmarks,CogapsResult-method (getPatternHallmarks), 25
getPatternMatrix, 25
getPatternMatrix,CogapsResult-method (getPatternMatrix), 25
getRetinaSubset, 26
getSampleFactors, 27
getSampleFactors,CogapsResult-method (getSampleFactors), 27
getSampleNames, 27
getSubsets, 28
getSubsets,CogapsResult-method (getSubsets), 28
getUnmatchedPatterns, 28
getUnmatchedPatterns,CogapsResult-method (getUnmatchedPatterns), 28
getValueOrRds, 29
getVersion, 29
getVersion,CogapsResult-method (getVersion), 29
GIST.data_frame, 30
GIST.matrix, 30
GIST.result, 30
GIST.uncertainty, 30
GWCoGAPS, 31

heatmap.2, 39

initialize,CogapsParams-method, 32
initialize,CogapsResult-method, 33
isRdsFile, 34

MANOVA, 34
MANOVA,matrix,CogapsResult-method (MANOVA), 34
modsimdata, 35

modsimresult, 35
ncolHelper, 35
nrowHelper, 36
parseExtraParams, 36
patternMarkers, 37
patternMarkers,CogapsResult-method (patternMarkers), 37
patternMatch, 37
plotPatternHallmarks, 38
plotPatternHallmarks,CogapsResult,list,numeric-method (plotPatternHallmarks), 38
plotPatternMarkers, 38
plotResiduals, 39
plotResiduals,CogapsResult-method (plotResiduals), 39
reconstructGene, 40
reconstructGene,CogapsResult-method (reconstructGene), 40
sampleUniformly, 41
sampleWithAnnotationWeights, 41
sampleWithExplicitSets, 42
scCoGAPS, 42
setAnnotationWeights, 44
setAnnotationWeights,CogapsParams-method (setAnnotationWeights), 44
setDistributedParams, 44
setDistributedParams,CogapsParams-method (setDistributedParams), 44
setFixedPatterns, 45
setFixedPatterns,CogapsParams-method (setFixedPatterns), 45
setParam, 46
setParam,CogapsParams-method (setParam), 46
startupMessage, 47
stitchTogether, 47
supported, 48
toCSV, 48
toCSV,CogapsResult,character-method (toCSV), 48