Package 'CAGEr'

October 8, 2015

Title Analysis of CAGE (Cap Analysis of Gene Expression) sequencing
data for precise mapping of transcription start sites and
promoterome mining

Version 1.10.1

Date 09-17-2015

Author Vanja Haberle, Department of Biology, University of Bergen, Norway <vanja.haberle@gmail.com>

Maintainer Vanja Haberle <vanja.haberle@gmail.com>

Imports utils, Rsamtools, GenomicRanges, IRanges, data.table,
beanplot, rtracklayer, som, VGAM

Depends methods, R (>= 2.15.0), BSgenome

Suggests BSgenome.Drerio.UCSC.danRer7, FANTOM3and4CAGE

Enhances parallel

Description Preprocessing of CAGE sequencing data, identification and normalization of transcription start sites and downstream analysis of transcription start sites clusters (promoters).

License GPL-3

biocViews Preprocessing, Sequencing, HighThroughputSequencing,
Transcription, Clustering, Visualization

Collate AllClasses.R GetMethods.R ImportFunctions.R ImportMethods.R
CorrelationMethods.R ExportFunctions.R ExportMethods.R
MergingMethods.R NormalizationFunctions.R
NormalizationMethods.R ExpressionProfilingMethods.R
ClusteringFunctions.R ClusteringMethods.R
CumulativeDistributionFunctions.R
CumulativeDistributionMethods.R QuantileWidthFunctions.R
QuantileWidthMethods.R AggregationFunctions.R
AggregationMethods.R ShiftingFunctions.R ShiftingMethods.R

NeedsCompilation no

R **topics documented:**

CAGEr-package	*Analysis of CAGE (Cap Analysis of Gene Expression) sequencing data for precise mapping of transcription start sites and promoterome mining*

Description

CAGEr package performs identification of transcription start sites and frequency of their usage from input CAGE sequencing data, normalization of raw CAGE tag count, clustering of TSSs into tag clusters (TC) and their aggregation across multiple CAGE experiments to construct the promoterome. It manipulates multiple CAGE experiments at once, performs expression profiling across experiments both at level of individual TSSs and clusters of TSSs, exports several different types of track files for visualization in the UCSC Genome Browser, performs analysis of promoter width and detects differential usage of TSSs (promoter shifting) between samples. Multicore option for parallel processing is supported on Unix-like platforms.

Details

Package:	CAGEr
Type:	Package
Version:	1.0
Date:	2012-12-18
License:	GPL-3
Depends:	R (>= 2.15.0), methods, BSgenome

Author(s)

Vanja Haberle

Maintainer: Vanja Haberle <vanja.haberle@bio.uib.no>

---

aggregateTagClusters	*Aggregating tag clusters across multiple CAGE datasets*

Description

Aggregates tag clusters (TCs) across all CAGE dataset within the CAGEset object to create a referent set of consensus clusters.

Usage

```
aggregateTagClusters(object, tpmThreshold = 5,
 excludeSignalBelowThreshold = TRUE,
 qLow = NULL, qUp = NULL, maxDist = 100)
```

Arguments

`object`	A [CAGEset](#) object
`tpmThreshold`	Only tag clusteres with normalized signal >= `tpmThreshold` will be used to construct consensus clusters.
`excludeSignalBelowThreshold`	
	When `TRUE` only tag clusteres with normalized signal >= `tpmThreshold` will contribute to the total CAGE signal of a consensus cluster, *i.e.* only the TCs that are used to construct consensus cluster. When set to `FALSE` all TCs that overlap consensus cluster will contribute to the total signal (regardless whether they pass the threshold or not), however only the TCs above the threshold will be used to define consensus cluster boundaries. Thus, it that case the TCs above the threshold are first used to construct consensus clusters and define their boundaries, but then CAGE signal from all TCs that fall within those boundaries is used to calculate total signal of a particular consensus cluster.
`qLow`	Position of which "lower" quantile should be used as 5' boundary of the tag cluster. If `qLow = NULL` start position of the TC is used. See Details.
`qUp`	Position of which "upper" quantile should be used as 3' boundary of the tag cluster. If `qUp = NULL` end position of the TC is used. `qUp` has to be >= `qLow`. See Details.
`maxDist`	Maximal length of the gap (in base-pairs) between two tag clusters for them to be part of the same consensus clusters. See Details.

Details

Tag clusters (TCs) returned by [clusterCTSS](#) function are constructed for every CAGE dataset within CAGEset object separatelly, based on the CAGE signal in that sample. Thus, TCs from two CAGE datasets can differ both in their number, genomic coordinates, position of dominant TSS and overall signal. To be able to compare all samples at the level of clusters of TSSs, TCs from all CAGE datasets are aggregated into a single set of consensus clusters. First, TCs with signal >= `tpmThreshold` from all CAGE datasets are selected, and their 5' and 3' boundaries are determined based on provided qLow and qUp parameters. If qLow = NULL and qUp = NULL the start and end coordinates, *i.e.* the full span of the TC is used, otherwise the positions of qLow and qUp quantiles are used as 5' and 3' boundary, respectively. Finally, the defined set of TCs from all CAGE datasets is reduced to a non-overlapping set of consensus clusters by merging overlapping TCs and TCs <= maxDist base-pairs apart. Consensus clusters represent a referent set of promoters that can be further used for expression profiling or detecting "shifting" (differentially used) promoters between different CAGE samples.

Value

The slots consensusClusters, tagClustersInConsensusClusters and consensusClustersTpmMatrix of the provided [CAGEset](#) object will be occupied by the genomic coordinates of consensus clusters, information on containing TCs and the total CAGE signal across all CAGE datasets, respectively.

Author(s)

Vanja Haberle

See Also

clusterCTSS

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

aggregateTagClusters(object = exampleCAGEset, tpmThreshold = 50,
excludeSignalBelowThreshold = FALSE, qLow = 0.1, qUp = 0.9, maxDist = 100)
```

CAGEset-class   *Class* "CAGEset"

Description

This class is used to store one or more CAGE (Cap Analysis of Gene Expression) datasets in the form of TSSs derived from CAGE tags and frequency of their usage, and to store and extract all information generated during the workflow.

Objects from the Class

Objects can be created by calls of the form new("CAGEset", ...).
Objects of the class contain information on the genomic coordinates of TSSs derived from sequenced CAGE tags from multiple experiments, number of tags supporting each TSS in each experiment, normalized CAGE signal at each TSS, all information on specified parameters and results of all downstream analyses. Object has to be created before reading in the data by specifying input files and their type, referent genome and labels for individual CAGE datasets, as described in the vignette. Data is read by applying a function to a created object and all further slots are filled during the workflow by applying specific functions.

Slots

genomeName: Object of class "character": the name of the BSgenome package used as the referent genome

inputFiles: Object of class "character": the paths to input files

inputFilesType: Object of class "character": the type of input files (e.g. bam)

sampleLabels: Object of class "character": the labels of individual CAGE experiments

librarySizes: Object of class "integer": the total number of CAGE tags per experiment

CTSScoordinates: Object of class "data.frame": the genomic coordinates of CAGE transcription start sites (CTSSs)

tagCountMatrix: Object of class "data.frame": the number of CAGE tags supporting every CTSS in each experiment

normalizedTpmMatrix: Object of class "data.frame": the normalized CAGE signal supporting every CTSS in each experiment

CTSSexpressionClusteringMethod: Object of class `"character"`: the method used for expression clustering of CTSSs

CTSSexpressionClasses: Object of class `"character"`: the labels of expression classes of CTSSs returned by expression clustering

clusteringMethod: Object of class `"character"`: the method used for clustering CTSSs into tag clusters (TC)

filteredCTSSidx: Object of class `"logical"`: the index of CTSSs included in tag clusters

tagClusters: Object of class `"list"`: the list of tag clusters per CAGE experiment

CTSScumulativesTagClusters: Object of class `"list"`: the cumulative distribution of CAGE signal along TCs

tagClustersQuantileLow: Object of class `"list"`: the positions of lower quantile(s) within TCs

tagClustersQuantileUp: Object of class `"list"`: the positions of upper quantile(s) within TCs

tagClustersInConsensusClusters: Object of class `"data.frame"`: the information on which TCs from which experiments are contained within each consensus cluster

consensusClusters: Object of class `"data.frame"`: the genomic coordinates of consensus clusters created by aggregating TCs across experiments

consensusClustersTpmMatrix: Object of class `"matrix"` the normalized CAGE signal for every consensus cluster in each experiment

consensusClustersExpressionClusteringMethod: Object of class `"character"` the method used for expression clustering of consensus clusters

consensusClustersExpressionClasses: Object of class `"character"` the labels of expression classes of consensus clusters returned by expression clustering

CTSScumulativesConsensusClusters: Object of class `"list"` the cumulative distribution of CAGE signal along consensus clusters

consensusClustersQuantileLow: Object of class `"list"` the positions of lower quantile(s) within consensus clusters

consensusClustersQuantileUp: Object of class `"list"` the positions of upper quantile(s) within consensus clusters

shiftingGroupX: Object of class `"character"` the label(s) of experiment(s) in the first shifting group

shiftingGroupY: Object of class `"character"` the label(s) of experiment(s) in the second shifting group

consensusClustersShiftingScores: Object of class `"data.frame"` the shifting scores and P-values/FDR for comparison of consensus clusters between two (groups of) experiments

Methods

CTSSclusteringMethod `signature(object = "CAGEset")`: extracts the method used for clustering CTSSs into tag clusters (TC)

CTSScoordinates `signature(object = "CAGEset")`: extracts the genomic coordinates of all CTSSs

CTSSnormalizedTpm `signature(object = "CAGEset")`: extracts the normalized CAGE signal supporting every CTSS in each experiment

CTSStagCount `signature(object = "CAGEset")`: extracts the number of CAGE tags support-
  ing every CTSS in each experiment

aggregateTagClusters `signature(object = "CAGEset")`: aggregates TCs across all experi-
  ments into consensus clusters

clusterCTSS `signature(object = "CAGEset")`: clusters CTSSs into TCs per experiment

consensusClusters `signature(object = "CAGEset")`: extracts the genomic coordinates and
  other information on consensus clusters

consensusClustersTpm `signature(object = "CAGEset")`: extracts the matrix with tpm values
  for consensus clusters across all samples

cumulativeCTSSdistribution `signature(object = "CAGEset")`: calculates the cumulative dis-
  tribution of CAGE signal along TCs or consensus clusters

exportCTSStoBedGraph `signature(object = "CAGEset")`: creates bedGraph files of CTSSs
  for visualization in the UCSC Genome Browser

exportToBed `signature(object = "CAGEset")`: creates various types of BED files for visual-
  ization in the UCSC Genome Browser

expressionClasses `signature(object = "CAGEset")`: extracts the labels of the expression classes
  of CTSSs or consensus clusters returned from expression profiling

extractExpressionClass `signature(object = "CAGEset")`: extracts CTSSs or consensus clus-
  ters belonging to specified expression class

genomeName `signature(object = "CAGEset")`: extracts the name of the BSgenome package
  used as the referent genome

getCTSS `signature(object = "CAGEset")`: reads in specified input files and fills in information
  on detected CTSSs and their tag count

getExpressionProfiles `signature(object = "CAGEset")`: performs expression clustering of
  CTSSs or consensus clusters across experiments

getShiftingPromoters `signature(object = "CAGEset")`: extracts consensus clusters with shift-
  ing score and/or FDR above specified threshold

inputFiles `signature(object = "CAGEset")`: extracts the paths of input CAGE data files

inputFilesType `signature(object = "CAGEset")`: extracts the type of input CAGE data files

librarySizes `signature(object = "CAGEset")`: extracts the library sizes of individual CAGE
  experiments within CAGEset object

mergeSamples `signature(object = "CAGEset", mergeIndex = "numeric")`: merges speci-
  fied experiments (samples) into one (e.g. replicas)

normalizeTagCount `signature(object = "CAGEset")`: normalizes raw CAGE tag count

plotCorrelation `signature(object = "CAGEset")`: plots pairwise scatter plots and calculates
  correlation between samples

plotExpressionProfiles `signature(object = "CAGEset")`: creates file with beanplots of expres-
  sion across experiments for CTSSs or consensus clusters belonging to different expression
  classes

plotInterquantileWidth `signature(object = "CAGEset")`: creates file with histograms of in-
  terquantile width

plotReverseCumulatives `signature(object = "CAGEset")`: creates file with reverse cumulative plots of CAGE tag count per CTSS

quantilePositions `signature(object = "CAGEset")`: calculates the positions of specified quantiles within TCs or consensus clusters

sampleLabels `signature(object = "CAGEset")`: extracts the labels of individual CAGE experiments within CAGEset object

scoreShift `signature(object = "CAGEset", groupX = "character", groupY = "character")`: calculates the shifting score and tests the statistical significance of differential TSS usage for consensus clusters between two specified (groups of) samples

setColors `signature(object = "CAGEset")`: assigns color to each sample to be used in visualisation

show `signature(object = "CAGEset")`: displays CAGEset object in a user friendly way

tagClusters `signature(object = "CAGEset")`: extracts the tag clusters for specified CAGE experiment

Coercion

`as(from, "CAGEset")`: Creates a CAGEset object from a data.frame object.

Author(s)

Vanja Haberle

Examples

```
showClass("CAGEset")
```

---

clusterCTSS	*Clustering CTSSs into tag clusters (TCs)*

Description

Clusters individual CAGE transcription start sites (CTSSs) along the genome into tag clusters using specified "ab initio" method, or assigns them to predefined genomic regions.

Usage

```
clusterCTSS(object, threshold = 1, nrPassThreshold = 1,
            thresholdIsTpm = TRUE, method = "distclu", maxDist = 20,
            removeSingletons = FALSE, keepSingletonsAbove = Inf,
            minStability = 1, maxLength = 500,
            reduceToNonoverlapping = TRUE, customClusters = NULL,
            useMulticore = FALSE, nrCores = NULL)
```

Arguments

`object`	A [`CAGEset`](#) object
`threshold, nrPassThreshold`	
	Only CTSSs with signal >= `threshold` in >= `nrPassThreshold` experiments will be used for clustering and will contribute towards total signal of the cluster.
`thresholdIsTpm`	Logical, is threshold raw tag count value (FALSE) or normalized signal (TRUE)
`method`	Method to be used for clustering. Can be one of the `"distclu"`, `"paraclu"` or `"custom"`. See Details.
`maxDist`	Maximal distance between two neighbouring CTSSs for them to be part of the same cluster. Used only when `method = "distclu"`, otherwise ignored.
`removeSingletons`	
	Logical, should tag clusters containing only one CTSS be removed. Ignored when `method = "custom"`.
`keepSingletonsAbove`	
	Controls which singleton tag clusters will be removed. When `removeSingletons = TRUE`, only singletons with signal < `keepSingletonsAbove` will be removed. Useful to prevent removing highly supported singleton tag clusters. Default value `Inf` results in removing all singleton TCs when `removeSingletons = TRUE`. Ignored when `removeSingletons = FALSE` or `method = "custom"`.
`minStability`	Minimal stability of the cluster, where stability is defined as ratio between maximal and minimal density value for which this cluster is maximal scoring. For definition of stability refer to Frith *et al.*, Genome Research, 2007. Clusters with stability < `minStability` will be discarded. Used only when `method = "paraclu"`, otherwise ignored.
`maxLength`	Maximal length of cluster in base-pairs. Clusters with length > `maxLength` will be discarded. Ignored when `method = "custom"`.
`reduceToNonoverlapping`	
	Logical, should smaller clusters contained within bigger cluster be removed to make a final set of tag clusters non-overlapping. Used only when `method = "paraclu"`. See Details.
`customClusters`	Genomic coordinates of predefined regions to be used to segment the CTSSs. It has to be a `data.frame` with following columns: `chr` (chromosome name), `start` (0-based start coordinate), `end` (end coordinate), `strand` (either `"+"`, or `"-"`). Used only when `method = "custom"`.
`useMulticore`	Logical, should multicore be used. `useMulticore = TRUE` is supported only on Unix-like platforms.
`nrCores`	Number of cores to use when `useMulticore = TRUE`. Default value `NULL` uses all detected cores.

Details

Two "ab initio" methods for clustering TSSs along the genome are supported: `"distclu"` and `"paraclu"`. `"distclu"` is an implementation of simple distance-based clustering of data attached to sequences, where two neighbouring TSSs are joined together if they are closer than some specified

distance. "paraclu" is an implementation of Paraclu algorithm for parametric clustering of data attached to sequences developed by M. Frith (Frith *et al.*, Genome Research, 2007, http://www.cbrc.jp/paraclu/). Since Paraclu finds clusters within clusters (unlike distclu), additional parameters (removeSingletons, keepSingletonsAbove, minStability, maxLength and reduceToNonoverlapping) can be specified to simplify the output by discarding too small (singletons) or too big clusters, and to reduce the clusters to a final set of non-overlapping clusters. Clustering is done for every CAGE dataset within CAGEset object separatelly, resulting in a different set of tag clusters for every CAGE dataset. TCs from different datasets can further be aggregated into a single referent set of consensus clusters by calling aggregateTagClusters function.

Value

The slots clusteringMethod, filteredCTSSidx and tagClusters of the provided CAGEset object will be occupied by the information on method used for clustering, CTSSs included in the clusters and list of tag clusters per CAGE experiment, respectively. To retrieve tag clusters for individual CAGE dataset use tagClusters function.

Author(s)

Vanja Haberle

References

Frith *et al.* (2007) A code for transcription initiation in mammalian genomes, *Genome Research* **18**(1):1-12, (http://www.cbrc.jp/paraclu/).

See Also

tagClusters
aggregateTagClusters

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

clusterCTSS(object = exampleCAGEset, threshold = 50, thresholdIsTpm = TRUE,
nrPassThreshold = 1, method = "distclu", maxDist = 20,
removeSingletons = TRUE, keepSingletonsAbove = 100)
```

---

consensusClusters   *Extracting consensus clusters from CAGEset object*

Description

Extracts the consensus clusters from a CAGEset object.

Usage

```
consensusClusters(object)
```

Arguments

object   A [CAGEset](#) object

Value

Returns a data.frame with information on consensus clusters, including genomic coordinates and total CAGE signal across all CAGE datasets.

Author(s)

Vanja Haberle

See Also

[tagClusters](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

clusters <- consensusClusters(exampleCAGEset)
head(clusters)
```

---

consensusClustersTpm   *Extracting consensus clusters tpm matrix from CAGEset object*

Description

Extracts the matrix with normalized CAGE tag values for consensus clusters across all samples from a CAGEset object.

Usage

```
consensusClustersTpm(object)
```

Arguments

object   A [CAGEset](#) object

Value

Returns a matrix with normalized CAGE tag values across all samples.

Author(s)

Vanja Haberle

See Also

[consensusClusters](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

clusters.tpm <- consensusClustersTpm(exampleCAGEset)
head(clusters.tpm)
```

CTSSclusteringMethod   *Extracting CTSS clustering method from CAGEset object*

Description

Extracts the label of the method used for CTSS clustering into tag clusters from a CAGEset object.

Usage

```
CTSSclusteringMethod(object)
```

Arguments

object   A [CAGEset](#) object

Value

Returns a label of the method used for CTSS clustering.

Author(s)

Vanja Haberle

See Also

[clusterCTSS](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

CTSSclusteringMethod(exampleCAGEset)
```

CTSScoordinates   *Extracting genomic coordinates of TSSs from CAGEset object*

Description

Extracts the genomic coordinates of all detected TSSs from a CAGEset object.

Usage

```
CTSScoordinates(object)
```

Arguments

object   A [CAGEset](#) object

Value

Returns a data.frame with genomic coordinates of all TSSs. pos column contains 1-based coordinate of the TSS.

Author(s)

Vanja Haberle

See Also

[CTSStagCount](#)
[CTSSnormalizedTpm](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

CTSS <- CTSScoordinates(exampleCAGEset)
head(CTSS)
```

CTSSnormalizedTpm   *Extracting normalized CAGE signal for TSSs from CAGEset object*

Description

Extracts the normalized CAGE signal for all detected TSSs in all CAGE datasets from a CAGEset object.

Usage

```
CTSSnormalizedTpm(object)
```

Arguments

object   A CAGEset object

Value

Returns a `data.frame` with normalized CAGE signal supporting each TSS (rows) in every CAGE dataset (columns).

Author(s)

Vanja Haberle

See Also

CTSScoordinates
CTSStagCount

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

CAGEsignal <- CTSSnormalizedTpm(exampleCAGEset)
head(CAGEsignal)
```

CTSStagCount   *Extracting CAGE tag count for TSSs from CAGEset object*

Description

Extracts the tag count for all detected TSSs in all CAGE datasets from a CAGEset object.

Usage

```
CTSStagCount(object)
```

Arguments

object   A CAGEset object

Value

Returns a `data.frame` with number of CAGE tags supporting each TSS (rows) in every CAGE dataset (columns).

Author(s)

Vanja Haberle

See Also

CTSScoordinates
CTSSnormalizedTpm

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

tagCount <- CTSStagCount(exampleCAGEset)
head(tagCount)
```

---

cumulativeCTSSdistribution

Calculating cumulative sum of CAGE signal along genomic region

---

Description

Calculates cumulative sum of CAGE signal along each tag cluster or consensus cluster in every CAGE dataset within CAGEset object.

Usage

```
cumulativeCTSSdistribution(object, clusters, useMulticore = FALSE,
                          nrCores = NULL)
```

Arguments

object	A CAGEset object
clusters	Which clusters should be used. Can be either clusters = "tagClusters" to calculate cumulative sum along tag clusters (different set of genomic coordinates for every CAGE experiment) or clusters = "consensusClusters" to calculate cumulative sum along consensus clusters (same set of genomic coordinates for every CAGE experiment).
useMulticore	Logical, should multicore be used. useMulticore = TRUE is supported only on Unix-like platforms.
nrCores	Number of cores to use when useMulticore = TRUE. Default value NULL uses all detected cores.

Value

The slot CTSScumulativesTagClusters (when clusters = "tagClusters") or CTSScumulativesConsensusClusters (when clusters = "consensusClusters") of the provided CAGEset object will be occupied by the list containing cumulative sum of the CAGE signal along genomic regions per CAGE experiment.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

cumulativeCTSSdistribution(object = exampleCAGEset, clusters = "tagClusters")
```

---

exampleCAGEset   *Example CAGEset object with zebrafish CAGE data*

Description

This is a [CAGEset](#) object that contains CAGE data for part of chromosome 13 from five different zebrafish (*Danio rerio*) samples. It is intended to be used as an input data in running examples from [CAGEr](#) package help pages.

Usage

```
data(exampleCAGEset)
```

Format

A [CAGEset](#) object

---

exportCTSStoBedGraph   *Creating bedGraph tracks of CAGE transcription starts sites*

Description

Creates bedGraph file(s) with track(s) of CAGE signal supporting each TSS that can be visualised in the UCSC Genome Browser.

Usage

```
exportCTSStoBedGraph(object, values, oneFile = TRUE)
```

Arguments

object	A [CAGEset](#) object
values	Specifies which values will be exported to the bedGraph file. Can be either "raw" to export raw tag count values or "normalized" to export normalized values.
oneFile	Logical, should all CAGE datasets be exported as individual tracks into the same bedGraph file (TRUE) or into separate bedGraph files (FALSE).

Value

Creates bedGraph file(s) in the working directory that can be directly visualised as custom tracks in the UCSC Genome Browser. If oneFile = TRUE one bedGraph file containing multiple annotated tracks will be created, otherwise two files per CAGE dataset will be created, one for plus strand and one for minus strand CTSSs, and they will be named according to the labels of individual datasets.

Author(s)

Vanja Haberle

See Also

[normalizeTagCount](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

exportCTSStoBedGraph(exampleCAGEset, values = "normalized", oneFile = TRUE)
```

---

exportToBed   *Creating BED tracks of TSSs and clusters of TSSs*

Description

Creates BED file(s) with track(s) of individual CTSSs, tag clusters or consensus clusters. CTSSs and consensus clusters can be optionally colored in the color of their expression class. Tag clusters and consensus clusters can be displayed in a gene-like representation with a line showing full span on the cluster, filled block showing interquantile range and a thick box denoting position of the dominant (most frequently used) TSS.

Usage

```
exportToBed(object, what, qLow = NULL, qUp = NULL,
            colorByExpressionProfile = FALSE, oneFile = TRUE)
```

Arguments

object	A [CAGEset](#) object
what	Which elements should be exported to BED track. Can be "CTSS" to export individual CTSSs, "tagClusters" to export tag clusters or "consensusClusters" to export consensus clusters. See Details.
qLow	Position of which "lower" quantile should be used as 5' boundary of the filled block in gene-like representation of the cluster. Default value NULL uses start position of the cluster. Ignored when what = "CTSS". See Details.

qUp   Position of which "upper" quantile should be used as 3' boundary of the filled
                      block in gene-like representation of the cluster. Default value `NULL` uses end
                      position of the cluster. Ignored when `what` = `"CTSS"`. See Details.

colorByExpressionProfile

                      Logical, should blocks be colored in the color of their corresponding expression
                      class. Ignored when `what` = `"tagClusters"`.

oneFile   Logical, should all CAGE datasets be exported as individual tracks into the
                      same BED file (TRUE) or into separate BED files (FALSE). Ignored when
                      `what` = `"CTSS"`, which by default produces only one track.

Details

This functions creates various representations of CTSSs, tag clusters and consensus clusters in the
BED format, which can be directly visualised in the UCSC Genome Browser.

When `what` = `"CTSS"`, one BED file with single track of 1bp blocks representing all detected
CTSSs (in all CAGE samples) is created. CTSSs can be colored according to their expression class
(provided the expression profiling of CTSSs was done by calling [getExpressionProfiles](#) func-
tion). Colors of expression classes match the colors in which they are shown in the plot returned
by the [plotExpressionProfiles](#) function. For colorByExpressionProfile = FALSE, CTSSs
included in the clusters are shown in black and CTSSs that were filtered out in gray.

When `what` = `"tagClusters"`, one track per CAGE dataset is created, which can be exported to a
single BED file (by setting oneFile = TRUE) or separate BED files (by setting oneFile = FALSE).
In case qLow = NULL and qUp = NULL, TCs are represented as simple blocks showing only the
full span of TC from the start to the end. Setting qLow and/or qUp parameters to a value of the
desired quantile creates a gene-like representation with a line showing full span of the TC, filled
block showing specified interquantile range and a thick 1bp block denoting position of the dominant
(most frequently used) TSS. All TCs in one track (one CAGE dataset) are shown in the same color.

When `what` = `"consensusClusters"` consensus clusters are exported to BED file. Since there
is only one set of consensus clusters common to all CAGE datasets, only one track is created in
case of a simple representation. This means that when qLow = NULL and qUp = NULL one track
with blocks showing the full span of consensus cluster from the start to the end is created. However,
the distribution of the CAGE signal within consensus cluster can be different in different CAGE
samples, resulting in different positions of quantiles and dominant TSS. Thus, when qLow and/or
qUp parameters are set to a value of the desired quantile, a separate track with a gene-like repre-
sentation is created for every CAGE dataset. These tracks can be exported to a single BED file (by
setting oneFile = TRUE) or separate BED files (by setting oneFile = FALSE). The gene-like repre-
sentation is analogous to the one described above for the TCs. In all cases consensus clusters can be
colored according to their expression class (provided the expression profiling of consensus clusters
was done by calling [getExpressionProfiles](#) function). Colors of expression classes match the
colors in which they are shown in the plot returned by the [plotExpressionProfiles](#) function. For
colorByExpressionProfile = FALSE all consensus clusters are shown in black.

Value

Creates BED file(s) in the working directory that can be directly visualised as custom tracks in the
UCSC Genome Browser.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

### exporting CTSSs colored by expression class
exportToBed(object = exampleCAGEset, what = "CTSS",
colorByExpressionProfile = TRUE)

### exporting tag clusters in gene-like representation
exportToBed(object = exampleCAGEset, what = "tagClusters",
qLow = 0.1, qUp = 0.9, oneFile = TRUE)
```

---

expressionClasses   *Extracting labels of expression classes*

Description

Retrieves labels of expression classes of either individual CTSSs or consensus clusters from a CAGEset object.

Usage

```
expressionClasses(object, what)
```

Arguments

object   A [CAGEset](#) object

what   Which level of expression clustering should be used. Can be either "CTSS" to ex-
                tract labels of expression classes of individual CTSSs or "consensusClusters"
                to extract labels of expression classes of consensus clusters.

Value

Returns character vector of labels of expression classes. The number of labels matches the number of expression clusters returned by [getExpressionProfiles](#) function.

Author(s)

Vanja Haberle

See Also

[getExpressionProfiles](#)
[plotExpressionProfiles](#)
[extractExpressionClass](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

exprClasses <- expressionClasses(exampleCAGEset, what = "CTSS")
exprClasses
```

---

extractExpressionClass

Extracting elements of the specified expression class

Description

Extracts CTSSs or consensus clusters belonging to a specified expression class.

Usage

```
extractExpressionClass(object, what, which = "all")
```

Arguments

object	A [CAGEset](#) object
what	Which level of expression clustering should be used. Can be either "CTSS" to extract expression class of individual CTSSs or "consensusClusters" to extract expression class of consensus clusters.
which	Which expression class should be extracted. It has to be one of the valid expression class labels (as returned by [expressionClasses](#) function), or "all" to extract members of all expression classes.

Value

Returns a data.frame of CTSSs (when what = "CTSS") or consensus clusters (when what = "consensusClusters") belonging to a specified expression class, with genomic coordinates and additional information. Last column contains the label of the corresponding expression class.

Author(s)

Vanja Haberle

See Also

[getExpressionProfiles](#)
[plotExpressionProfiles](#)
[expressionClasses](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

CTSSexprClasses <- extractExpressionClass(exampleCAGEset, what = "CTSS",
which = "all")
head(CTSSexprClasses)
```

FANTOM5humanSamples   *List of FANTOM5 human CAGE samples*

Description

This is a `data.frame` object that contains information about all human primary cell, cell line and tissue CAGE datasets (988 datasets) produced by FANTOM5 constortium, which are available for import with `CAGEr`. The list containes all samples published in the main FANTOM5 publication (Forrest *et al.* Nature 2014) as presented in the Supplementary Table 1. The columns provide the following information:

`sample`: name/label of individual sample, which can be used as `sample` argument in the `importPublicData` function to retrieve specified sample.

`type`: type of the sample, either `"cell line"`, `"primary cell"` or `"tissue"`.

`description`: description of individual sample as provided by the FANTOM5 consortium.

`library_id`: unique ID of the CAGE library as provided by the FANTOM5 consortium.

`data_url`: URL to gzip-ed TSS file at online FANTOM5 data resource, which is used by `importPublicData` to fetch given sample.

Usage

```
data(FANTOM5humanSamples)
```

Format

A `data.frame` object

FANTOM5mouseSamples   *List of FANTOM5 mouse CAGE samples*

Description

This is a `data.frame` object that contains information about all mouse primary cell, cell line and tissue CAGE datasets (395) produced by FANTOM5 constortium, which are available for import with `CAGEr`. The list containes all samples published in the main FANTOM5 publication (Forrest *et al.* Nature 2014) as presented in the Supplementary Table 1. The columns provide the following information:

`sample`: name/label of individual sample, which can be used as `sample` argument in the `importPublicData` function to retrieve specified sample.

type: type of the sample, either "cell line", "primary cell" or "tissue".
description: description of individual sample as provided by the FANTOM5 consortium.
library_id: unique ID of the CAGE library as provided by the FANTOM5 consortium.
data_url: URL to gzip-ed TSS file at online FANTOM5 data resource, which is used by importPublicData
to fetch given sample.

Usage

```
data(FANTOM5humanSamples)
```

Format

A data.frame object

---

genomeName   *Extracting genome name from CAGEset object*

Description

Extracts the name of a referent genome from a CAGEset object.

Usage

```
genomeName(object)
```

Arguments

object   A CAGEset object

Value

Returns a name of a BSgenome package used as a referent genome.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

genomeName(exampleCAGEset)
```

---

getCTSS   *Reading CAGE data from input file(s) and detecting TSSs*

Description

Reads input CAGE datasets into CAGEset object, constructs CAGE transcriptions start sites (CTSSs) and counts number of CAGE tags supporting every CTSS in each input experiment. Preprocessing and quality filtering of input CAGE tags, as well as correction of CAGE-specific 'G' nucleotide addition bias can be also performed before constructing TSSs.

Usage

```
getCTSS(object, sequencingQualityThreshold = 10,
        mappingQualityThreshold = 20, removeFirstG = TRUE,
        correctSystematicG = TRUE)
```

Arguments

object   A CAGEset object

sequencingQualityThreshold, mappingQualityThreshold

Only CAGE tags with average sequencing quality >= sequencingQualityThreshold and mapping quality >= mappingQualityThreshold are kept. Used only if inputFileType(object) == "bam", *i.e* when input files are BAM files of aligned sequenced CAGE tags, otherwise ignored. If there are no sequencing quality values in the BAM file (*e.g.* HeliScope single molecule sequencer does not return sequencing qualities) all reads will by default have this value set to -1. Since the default value of sequencingQualityThreshold is 10, all the reads will consequently be discarded. To avoid this behaviour and keep all sequenced reads set sequencingQualityThreshold to -1 when processing data without sequencing qualities. If there is no information on mapping quality in the BAM file (*e.g.* software used to align CAGE tags to the referent genome does not provide mapping quality) the mappingQualityThreshold parameter is ignored.

removeFirstG   Logical, should the first nucleotide of the CAGE tag be removed in case it is a G and it does not map to the referent genome (*i.e.* it is a mismatch). See Details.

correctSystematicG

Logical, should the systematic correction of the first G nucleotide be performed for the positions where there is a G in the CAGE tag and G in the genome. This step is performed in addition to removing the first G of the CAGE tags when it is a mismatch, *i.e.* this option can only be used when removeFirstG = TRUE, otherwise it is ignored. The frequency of adding a G to CAGE tags is estimated from mismatch cases and used to systematically correct the G addition for positions with G in the genome. See Details.

Details

In the CAGE experimental protocol an additional G nucleotide is often attached to the 5' end of the tag by the template-free activity of the reverse transcriptase used to prepare cDNA (Harbers

and Carninci, Nature Methods 2005). In cases where there is a G at the 5' end of the CAGE tag that does not map to the corresponding genome sequence, it can confidently be considered spurious and should be removed from the tag to avoid misannotating actual TSS. Thus, setting removeFirstG = TRUE is highly recommended.

However, when there is a G both at the beginning of the CAGE tag and in the genome, it is not clear whether the original CAGE tag really starts at this position or the G nucleotide was added later in the experimental protocol. To systematically correct CAGE tags mapping at such positions, a general frequency of adding a G to CAGE tags can be calculated from mismatch cases and applied to estimate the number of CAGE tags that have G added and should actually start at the next nucleotide/position. The option correctSystematicG is an implementation of the correction algorithm described in Carninci *et al.*, Nature Genetics 2006, Supplementary Information section 3-e.

Value

The slots librarySizes, CTSScoordinates and tagCountMatrix of the provided [CAGEset](#) object will be occupied by the information on CTSSs created from input CAGE files.

Author(s)

Vanja Haberle

References

Harbers and Carninci (2005) Tag-based approaches for transcriptome research and genome annotation, *Nature Methods* **2**(7):495-502.
Carninci *et al.* (2006) Genome-wide analysis of mammalian promoter architecture and evolution, *Nature Genetics* **38**(7):626-635.

See Also

[CTSScoordinates](#)
[CTSStagCount](#)

Examples

```
library(BSgenome.Drerio.UCSC.danRer7)

pathsToInputFiles <- system.file("extdata", c("Zf.unfertilized.egg.chr17.ctss",
"Zf.30p.dome.chr17.ctss", "Zf.prim6.rep1.chr17.ctss"), package="CAGEr")
labels <- paste("sample", seq(1,3,1), sep = "")

myCAGEset <- new("CAGEset", genomeName = "BSgenome.Drerio.UCSC.danRer7",
inputFiles = pathsToInputFiles, inputFilesType = "ctss", sampleLabels = labels)

getCTSS(myCAGEset)
```

---

getExpressionProfiles   *CAGE data based expression clustering*

Description

Performs clustering of CAGE derived expression across multiple experiments, both at level of individual TSSs or entire clusters of TSSs.

Usage

```
getExpressionProfiles(object, what, tpmThreshold = 5,
                      nrPassThreshold = 1, method = "som",
                      xDim = 5, yDim = 5)
```

Arguments

object   A [CAGEset](#) object

what   At which level should the expression clustering be done. Can be either "CTSS" to perform clustering of individual CTSSs or "consensusClusters" to perform clustering of consensus clusters. See Details.

tpmThreshold, nrPassThreshold

                Only CTSSs or consensus clusters (depending on what parameter) with normalized CAGE signal >= tpmThreshold in >= nrPassThreshold experiments will be included in expression clustering.

method   Method to be used for expression clustering. Can be either "som" to use the self-organizing map (SOM) algorithm (Toronen *et al*., FEBS Letters 1999) implemented in the the som function from som package, or "kmeans" to use the K-means algorithm implemented in the kmeans function from stats package.

xDim, yDim   When method = "kmeans", xDim specifies number of clusters that will be returned by K-means algorithm and yDim is ignored. When method = "som", xDim specifies the the first and yDim the second dimension of the self-organizing map, which results in total xDim * yDim clusters returned by SOM.

Details

Expression clustering can be done at level of individual CTSSs, in which case the feature vector used as input for clustering algorithm contains log-transformed and scaled (divided by standard deviation) normalized CAGE signal at individual TSS across multiple experiments. Only TSSs with normalized CAGE signal >= tpmThreshold in at least nrPassThreshold CAGE experiments are used for expression clustering. However, CTSSs along the genome can be spatially clustered into tag clusters for each experiment separately using the [clusterCTSS](#) function, and then aggregated across experiments into consensus clusters using [aggregateTagClusters](#) function. Once the consensus clusters have been created, expression clustering at the level of these wider genomic regions (representing entire promoters rather than individual TSSs) can be performed. In that case the feature vector used as input for clustering algorithm contains normalized CAGE signal within entire consensus cluster across multiple experiments, and threshold values in tpmThreshold and nrPassThreshold are applied to entire consensus clusters.

Value

If what = "CTSS" the slots CTSSexpressionClusteringMethod and CTSSexpressionClasses will be occupied, and if what = "consensusClusters" the slots consensusClustersExpressionClusteringMethod and consensusClustersExpressionClasses of the provided [CAGEset](#) object will be occupied with the results of expression clustering. Labels of expression classes (clusters) can be retrieved using [expressionClasses](#) function, and elements belonging to a specific expression class can be selected using [extractExpressionClass](#) function.

Author(s)

Vanja Haberle

References

Toronen *et al.* (1999) Analysis of gene expression data using self-organizing maps, *FEBS Letters* **451**:142-146.

See Also

[plotExpressionProfiles](#)
[expressionClasses](#)
[extractExpressionClass](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

getExpressionProfiles(object = exampleCAGEset, what = "CTSS",
tpmThreshold = 50, nrPassThreshold = 1, method = "som", xDim = 3, yDim = 3)
```

---

getShiftingPromoters   *Selecting consensus clusters with shifting score above specified threshold*

Description

Extracts consensus clusters with shifting score and/or FDR (adjusted P-value from Kolmogorov-Smirnov test) above specified threshold. Returns their genomic coordinates, total CAGE signal and the position of dominant TSS in the two compared groups of CAGE samples, along with the value of the shifting score, P-value and FDR. Scores and P-values/FDR have to be calculated beforehand by calling [scoreShift](#) function.

Usage

```
getShiftingPromoters(object, tpmThreshold = 0, scoreThreshold = -Inf, fdrThreshold = 1)
```

Arguments

object	A [CAGEset](#) object
tpmThreshold	Consensus clusters with total CAGE signal >= tpmThreshold in each of the compared groups will be returned.
scoreThreshold	Consensus clusters with shifting score >= scoreThreshold will be returned. The default value -Inf returns all consensus clusters (for which score could be calculated, *i.e.* the ones that have at least one tag in each of the comapred samples).
fdrThreshold	Consensus clusters with adjusted P-value (FDR) from Kolmogorov-Smirnov test >= fdrThreshold will be returned. The default value 1 returns all consensus clusters (for which K-S test could be performed, *i.e.* the ones that have at least one tag in each of the compared samples).

Value

Returns a data.frame of shifting promoters with genomic coordinates and positions of dominant TSS and CAGE signal in the two compared (groups of) samples, along with shifting score and adjusted P-value (FDR).

Author(s)

Vanja Haberle

See Also

[scoreShift](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

shifting.promoters <- getShiftingPromoters(object = exampleCAGEset,
tpmThreshold = 100, scoreThreshold = 0.4, fdrThreshold = 0.01)
head(shifting.promoters)
```

---

importPublicData	*Importing publicly available CAGE data from various resources*

Description

Imports CAGE data from different sources into a [CAGEset](#) object. After the CAGEset object has been created the data can be further manipulated and visualized using other functions available in the *CAGEr* package and integrated with other analyses in R. Available resources include:
- FANTOM5 datasets (Forrest *et al.* Nature 2014) for numerous human and mouse samples (primary cells, cell lines and tissues), which are fetched directly from FANTOM5 online resource.
- FANTOM3 and 4 datasets (Carninci *et al.* Science 2005, Faulkner *et al.* Nature Genetics 2009,

Suzuki *et al.* Nature Genetics 2009) from *FANTOM3and4CAGE* data package available from Bioconductor
- ENCODE datasets (Djebali *et al.* Nature 2012) for numerous human cell lines from *ENCODEprojectCAGE* data package, which is available for download from http://promshift.genereg.net/CAGEr/.
- Zebrafish developmental timecourse datasets (Nepal *et al.* Genome Research 2013) from *ZebrafishDevelopmentalCAGE* data package, which is available for download from http://promshift.genereg.net/CAGEr/.

Usage

```
importPublicData(source, dataset, group, sample)
```

Arguments

source
: Character vector specifying one of the available resources for CAGE data. Can be one of the following:
  `"FANTOM5"`: for fetching and importing CAGE data for various human or mouse primary cells, cell lines and tissues from the online FANTOM5 resource (http://fantom.gsc.riken.jp/5/data/). All data published in main FANTOM5 publication by Forrest *et al.* is available.
  `"FANTOM3and4"`: for importing CAGE data for various human or mouse tissues produced within FANTOM3 and FANTOM4 projects. Requires data package *FANTOM3and4CAGE* to be installed. This data package is available from Bioconductor.
  `"ENCODE"`: for importing CAGE data for human cell lines from ENCODE project published by Djebali *et al.*. Requires data package *ENCODEprojectCAGE* to be installed. This data package is available for download from http://promshift.genereg.net/CAGEr/.
  `"ZebrafishDevelopment"`: for importing CAGE data from developmental timecourse of zebrafish (*Danio rerio*) published by Nepal *et al.*. Requires data package *ZebrafishDevelopmentalCAGE* to be installed. This data package is available for download from http://promshift.genereg.net/CAGEr/.
  See Details for further explanation of individual resources.

dataset
: Character vector specifying one or more of the datasets available in the selected resource. For FANTOM5 it can be either `"human"` or `"mouse"`, and only one of them can be specified at a time. For other resources please refer to the vignette of the corresponding data package for the list of available datasets. Multiple datasets mapped to the same genome can be specified to combine selected samples from each.

group
: Character string specifying one or more groups within specified dataset(s), from which the samples should be selected. group argument is used only when importing TSSs from data packages and ignored when source=`"FANTOM5"`. For available groups in each dataset please refer to the vignette of the corresponding data package. Either only one group has to be specified (if all selected samples belong to the same group) or one group per sample (if samples belong to different groups). In the latter case, the number of elements in group must match the number of elements in sample.

sample
: Character string specifying one or more CAGE samples. Check the corresponding data package for available samples within each group and their labels. For FANTOM5 resource, list of all human (~1000) and mouse (~) samples can be obtained in *CAGEr* by loading data(FANTOM5humanSamples) and

data(FANTOM5mouseSamples), respectively. Use the names from the sample column to specify which samples should be imported.

Details

CAGE data from different sources is available for importing directly into CAGEset object for further manipulation with *CAGEr*.

FANTOM consortium provides single base-pair resolution TSS data for numerous human and mouse primary cells, cell lines and tissues produced within FANTOM5 project (Forrest *et al.* Nature 2014). These are directly fetched from their online resource at http://fantom.gsc.riken.jp/5/data and imported into a CAGEset object. To use this resource specify source="FANTOM5". The dataset argument can be either "human" or "mouse", but not both at the same time. The list of all human and mouse samples can be obtained by loading data(FANTOM5humanSamples) and data(FANTOM5mouseSamples). The sample column gives the names of individual samples that should be provided as sample argument. See example below.

TSS data from previous FANTOM3 and FANTOM4 projects (Carninci *et al.*, Faulkner *et al.*, Suzuki *et al.*) are also available through *FANTOM3and4CAGE* data package. This data package can be installed directly from Bioconductor. To use this resource install and load *FANTOM3and4CAGE* package and specify source="FANTOM3and4". The dataset argument can be a name of any of the datasets available in this package. Load data(FANTOMhumanSamples) or data(FANTOMmouseSamples) for the list of available datasets with group and sample labels for specific human or mouse samples. These have to be provided as dataset, group and sample arguments to import selected samples. If all samples belong to the same group, only this one group can be provided, otherwise, for each sample a corresponding group has to be specified, *i.e.* the number of elements in group must match the numer of elements in sample.

ENCODE consortium produced CAGE data for numerous human cell lines (Djebali *et al.* Nature 2012). We have used these data to derive single base-pair resolution TSSs and collected them into an R data package *ENCODEprojectCAGE*. This data package is available for download from http://promshift.genereg.net/CAGEr/. To use this resource install and load *ENCODEprojectCAGE* data package and specify source="ENCODE". The dataset argument can be a name of any of the datasets available in this package. Load data(ENCODEhumanCellLinesSamples) for the list of available datasets with group and sample labels for specific samples. These have to be provided as dataset, group and sample arguments to import selected samples. Multiple datasets can be combined together, by specifying them as dataset argument. If all samples belong to the same dataset and the same group, these dataset and group can be specified only once, otherwise, for each sample a corresponding dataset and group have to be specified, *i.e.* the number of elements in dataset and group must match the numer of elements in sample.

Precise TSSs are also available for zebrafish (*Danio Rerio*) from CAGE data published by Nepal *et al.* for 12 developmental stages. These have been collected into a data package *ZebrafishDevelopmentalCAGE*, which is available for download from http://promshift.genereg.net/CAGEr/. To use this resource install and load *ZebrafishDevelopmentalCAGE* data package and specify source="ZebrafishDevelopment". Load data(ZebrafishSamples) for the list of available datasets and group and sample labels, which have to be specified to import these data.

Value

A CAGEset object is returned. Slots librarySizes, CTSScoordinates and tagCountMatrix are occupied by the single base-pair resolution TSS data imported from the selected resource.

Author(s)

Vanja Haberle

References

Carninci *et al.* (2005) The Transcriptional Landscape of the Mammalian Genome, *Science* **309**(5740):1559-1563.
Djebali *et al.* (2012) Landscape of transcription in human cells, *Nature* **488**(7414):101-108.
Faulkner *et al.* (2009) The regulated retrotransposon transcriptome of mammalian cells, *Nature Genetics* **41**:563-571.
Forrest *et al.* (2014) A promoter-level mammalian expression atlas, *Nature* **507**(7493):462-470.
Nepal *et al.* (2013) Dynamic regulation of the transcription initiation landscape at single nucleotide resolution during vertebrate embryogenesis, *Genome Research* **23**(11):1938-1950.
Suzuki *et al.* (2009) The transcriptional network that controls growth arrest and differentiation in a human myeloid leukemia cell line, *Nature Genetics* **41**:553-562.

See Also

[getCTSS](#)

Examples

```
### importing FANTOM5 data

# list of FANTOM5 human tissue samples
data(FANTOM5humanSamples)
head(subset(FANTOM5humanSamples, type == "tissue"))

# import selected samples
exampleCAGEset <- importPublicData(source="FANTOM5", dataset = "human", sample = c("adipose_tissue__adult__pool1

exampleCAGEset


### importing FANTOM3/4 data from a data package
library(FANTOM3and4CAGE)

# list of mouse datasets available in this package
data(FANTOMmouseSamples)
unique(FANTOMmouseSamples$dataset)
head(subset(FANTOMmouseSamples, dataset == "FANTOMtissueCAGEmouse"))
head(subset(FANTOMmouseSamples, dataset == "FANTOMtimecourseCAGEmouse"))

# import selected samples from two different mouse datasets
exampleCAGEset <- importPublicData(source="FANTOM3and4", dataset = c("FANTOMtissueCAGEmouse", "FANTOMtimecourse(

exampleCAGEset
```

inputFiles   *Extracting paths to input files from CAGEset object*

Description

Extracts the paths to CAGE data input files from a CAGEset object.

Usage

```
inputFiles(object)
```

Arguments

object   A [CAGEset](#) object

Value

Returns a character vector of paths to CAGE data input files.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

inputFiles(exampleCAGEset)
```

inputFilesType   *Extracting type of input files from CAGEset object*

Description

Extracts the information on the type of CAGE data input files from a CAGEset object.

Usage

```
inputFilesType(object)
```

Arguments

object   A [CAGEset](#) object

Value

Returns the label of the file type of CAGE data input files, *e.g.* "bam" or "ctss".

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

inputFilesType(exampleCAGEset)
```

---

librarySizes *Extracting library sizes from CAGEset object*

Description

Extracts the library sizes (total number of CAGE tags) for all CAGE datasets from a CAGEset object.

Usage

```
librarySizes(object)
```

Arguments

object   A [CAGEset](#) object

Value

Returns an integer vector of total number of CAGE tags (library size) for all CAGE datasets in the CAGEset object.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

librarySizes(exampleCAGEset)
```

---

mergeCAGEsets   *Merging two CAGEset objects into one*

Description

Merges two CAGEset objects into one by combining the CTSS genomic coordinates and raw tag counts. Merged CAGEset object will contain a union of TSS positions present in the two input CAGEset objects and raw tag counts for those TSSs in all samples from both input CAGEset objects.

Usage

```
mergeCAGEsets(cs1, cs2)
```

Arguments

cs1   A CAGEset object

cs2   A CAGEset object

Details

Note that merging discards all other information present in the two CAGEset objects, *i.e.* the merged object will not contain any normalised tag counts, CTSS clusters, quantile positions, *etc.*, so these have to be calculated again by calling the appropriate functions on the merged CAGEset object. Also, it is only possible to merge two CAGEset objects that contain TSS information for the same reference genome and do not share any sample names.

Value

A CAGEset object is returned, which contains a union of TSS positions present in the two input CAGEset objects and raw tag counts for those TSSs in all samples from both input CAGEset objects.

Author(s)

Vanja Haberle

See Also

CAGEset

Examples

```
library(BSgenome.Drerio.UCSC.danRer7)

pathsToInputFiles <- system.file("extdata", c("Zf.unfertilized.egg.chr17.ctss",
"Zf.30p.dome.chr17.ctss", "Zf.prim6.rep1.chr17.ctss"), package="CAGEr")

myCAGEset1 <- new("CAGEset", genomeName = "BSgenome.Drerio.UCSC.danRer7",
inputFiles = pathsToInputFiles[1:2], inputFilesType = "ctss", sampleLabels =
```

```
c("sample1", "sample2"))
getCTSS(myCAGEset1)

myCAGEset2 <- new("CAGEset", genomeName = "BSgenome.Drerio.UCSC.danRer7",
inputFiles = pathsToInputFiles[3], inputFilesType = "ctss", sampleLabels =
"sample3")
getCTSS(myCAGEset2)

myCAGEset <- mergeCAGEsets(myCAGEset1, myCAGEset2)
```

---

mergeSamples   *Merging CAGE datasets (samples)*

Description

Merges individual CAGE datasets (experiments, samples) within the CAGEset object into specified groups.

Usage

```
mergeSamples(object, mergeIndex, mergedSampleLabels)
```

Arguments

object   A [CAGEset](#) object

mergeIndex   Integer vector specifying which experiments should be merged. Must be the same length as the number of datasets in the CAGEset object. See Details.

mergedSampleLabels

Labels for the merged datasets. Must be the same lenght as the number of unique values in mergeIndex. See Details.

Details

This function merges CAGE datasets within the CAGEset object at the level of CTSS tag counts, *i.e.* tag counts of individual CTSS are summed over a group of datasets that are being merged. After merging, all other slots in the CAGEset object will be reset and any previous data for individual experiments will be removed.

mergeIndex controls which datasets will be merged. It is an integer vector that assigns a value to each dataset in the CAGEset object in the same order as they are returned by sampleLabels(object). Datasets with the same value in the mergeIndex will be merged. For example, if there are 8 CAGE datasets in the CAGEset object and mergeIndex = c(1,1,2,2,3,2,4,4), this will merge a) samples 1 and 2 b) samples 3, 4 and 6 c) samples 7 and 8, and it will leave sample 5 as it is, resulting in 4 final merged datasets.

Labels provided in mergedSampleLabels will be assigned to merged datasets in the ascending order of mergeIndex values, *i.e.* first label will be assigned to a dataset created by merging datasets labeled with lowest mergeIndex value (in this case 1), *etc*.

Value

The slots `sampleLabels`, `librarySizes` and `tagCountMatrix` of the provided [CAGEset](#) object will be updated with the information on merged CAGE datasets and will replace the previous information on individual CAGE datasets. All further slots with downstream information will be reset.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

mergeSamples(exampleCAGEset, mergeIndex = c(1,1,2),
mergedSampleLabels = c("mergedSample1", "mergedSample2"))
exampleCAGEset
```

---

normalizeTagCount   *Normalizing raw CAGE tag count*

Description

Normalizes raw CAGE tag count per CTSS in all experiments to a same referent distribution. A simple tag per million normalization or normalization to a referent power-law distribution (Balwierz *et al*., Genome Biology 2009) can be specified.

Usage

```
normalizeTagCount(object, method = "powerLaw", fitInRange = c(10, 1000),
                  alpha = 1.25, T = 10^6)
```

Arguments

object	A [CAGEset](#) object
method	Method to be used for normalization. Can be either `"simpleTpm"` to convert tag counts to tags per million or `"powerLaw"` to normalize to a referent power-law distribution, or `"none"` to keep using the raw tag counts in downstream analyses.
fitInRange	An integer vector with two values specifying a range of tag count values to be used for fitting a power-law distribution to reverse cumulatives. Used only when `method = "powerLaw"`, otherwise ignored. See Details.
alpha	$-1 *$ alpha will be the slope of the referent power-law distribution in the log-log representation. Used only when `method = "powerLaw"`, otherwise ignored. See Details.
T	Total number of CAGE tags in the referent power-law distribution. Setting `T = 10^6` results in normalized values that correspond to tags per million in the referent distribution. Used only when `method = "powerLaw"`, otherwise ignored. See Details.

Details

It has been shown that many CAGE datasets follow a power-law distribution (Balwierz *et al.*, Genome Biology 2009). Plotting the number of CAGE tags (X-axis) against the number of TSSs that are supported by >= of that number of tags (Y-axis) results in a distribution that can be approximated by a power-law. On a log-log scale this theoretical referent distribution can be described by a monotonically decreasing linear function y = -1 * alpha * x + beta, which is fully determined by the slope alpha and total number of tags T (which together with alpha determines the value of beta). Thus, by specifying parameters alpha and T a desired referent power-law distribution can be selected. However, real CAGE datasets deviate from the power-law in the areas of very low and very high number of tags, so it is advisable to discard these areas before fitting a power-law distribution. fitInRange parameter allows to specify a range of values (lower and upper limit of the number of CAGE tags) that will be used to fit a power-law. Plotting reverse cumulatives using plotReverseCumulatives function can help in choosing the best range of values. After fitting a power-law distribution to each CAGE dataset individually, all datasets are normalized to a referent distribution specified by alpha and T. When T = 10^6, normalized values are expressed as tags per million (tpm).

Value

The slot normalizedTpmMatrix of the provided CAGEset object will be occupied by normalized CAGE signal values per CTSS across all experiments, or with the raw tag counts (in case method = "none").

Author(s)

Vanja Haberle

References

Balwierz *et al.* (2009) Methods for analyzing deep sequencing expression data: constructing the human and mouse promoterome with deepCAGE data, *Genome Biology* **10**(7):R79.

See Also

plotReverseCumulatives

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

normalizeTagCount(exampleCAGEset, method = "powerLaw")
```

---

plotCorrelation	*Plotting pairwise scatter plots of CAGE signal and calculating correlation between samples*

Description

Creates PNG file with scatter plots of CAGE signal for all pairs of selected samples and calculates the correlation between them.

Usage

```
plotCorrelation(object, what = "CTSS", values = "raw",
samples = "all", method = "pearson",
tagCountThreshold = 1, applyThresholdBoth = FALSE)
```

Arguments

object	A [CAGEset](#) object
what	Which level should be used for plotting and calculating correlation. Can be either "CTSS" to use individual TSSs or "consensusClusters" to use consensus clusters, *i.e.* entire promoters.
values	Specifies which values will be used for plotting and calculating correlation. Can be either "raw" to use raw tag count per TSS or "normalized" to use normalized CAGE signal. Used only when what = "CTSS", otherwise ignored.
samples	Character vector of sample labels for which the scatter plots should be plotted and correlation calculated. Can be either "all" to plot and calculate pairwise correlations between all samples in a [CAGEset](#) object, or a subset of valid sample labels as returned by [sampleLabels](#) function.
method	A character string indicating which correlation coefficient should be computed. Passed to cor function. Can be one of "pearson", "spearman", or "kendall".
tagCountThreshold, applyThresholdBoth	
	Only TSSs with tag count >= tagCountThreshold in either one (applyThresholdBoth = FALSE) or both samples (applyThresholdBoth = TRUE) are plotted and used to calculate correlation.

Value

Creates PNG file named "Pairwise_tag_count_correlation.png" in the working directory. Returns a matrix of pairwise correlations between selected samples.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

plotCorrelation(object = exampleCAGEset)
```

---

plotExpressionProfiles

Plotting expression profiles derived from CAGE data

Description

Creates PDF file with beanplots representing distribution of normalized expression across CAGE experiments for individual expression classes. Different expression classes are shown in different colors and are labeled according to the labels returned by expression clustering.

Usage

```
plotExpressionProfiles(object, what)
```

Arguments

object	A [CAGEset](#) object
what	Which level of expression clustering should be used. Can be either "CTSS" to plot expression profiles of individual CTSSs or "consensusClusters" to plot expression profiles of consensus clusters.

Details

The created file contains beanplots representing distribution of normalized expression across CAGE experiments for individual expression classes shown in separate boxes. Each labeled box represents one expression class and contains a set of beanplots - one per CAGE experiment. Individual CAGE experiments are shown on X-axis and scaled normalized expression on Y-axis. Individual beanplots show distribution of normalized expression values of elements belonging to specific expression class in particular CAGE experiment, and the entire box represents single expression profile. Different expression classes (boxes) are plotted in different colors and are labeled with labels returned by expression clustering.

Value

Creates PDF file named "CTSS_expression_profiles.pdf" (in case what = "CTSS") or "consensus-Clusters_expression_profiles.pdf" (in case what = "consensusClusters") in the working directory.

Author(s)

Vanja Haberle

See Also

getExpressionProfiles
expressionClasses
extractExpressionClass

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

plotExpressionProfiles(object = exampleCAGEset, what = "CTSS")
```

---

plotInterquantileWidth

Plotting distribution of interquantile width

Description

Creates PDF file with histograms showing distribution of interquantile width of tag clusters or consensus clusters in each CAGE dataset.

Usage

```
plotInterquantileWidth(object, clusters, tpmThreshold = 5,
                  qLow = 0.1, qUp = 0.9, xlim = c(0,150), ...)
```

Arguments

object	A CAGEset object
clusters	Which clusters should be used. Can be either "tagClusters" to plot distribution of interquantile width of tag clusters or "consensusClusters" to plot distribution of interquantile width of consensus clusters.
tpmThreshold	Only clusters with normalized signal >= tpmThreshold will be included in the histogram.
qLow	Position of which "lower" quantile should be used as 5' boundary. See Details.
qUp	Position of which "upper" quantile should be used as 3' boundary. See Details.
xlim	The x axis limits of the plot.
...	Additional arguments passed to plot() function, such as ylim, *etc.*.

Details

Interquantile width is a width (in base-pairs) of the central part of the genomic region (bounded by the positions of specified qLow and qUp quantiles) that contains >= (qUp - qLow) * 100% of the CAGE signal. Positions of specified quantiles within each cluster have to be calculated beforehand by calling quantilePositions function. Interquantile width is a more robust measure of the promoter width than the total span of the region, because it takes into account the magnitude of the expression in the region.

Value

Creates PDF file named "tagClusters_interquantile_width_all_samples.pdf" or "consensusClusters_interquantile_width_all_s
in the working directory (depending on the value of cluster parameter). The file contains his-
tograms showing distribution of interquantile width in every CAGE experiment.

Author(s)

Vanja Haberle

See Also

quantilePositions

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

plotInterquantileWidth(object = exampleCAGEset, clusters = "tagClusters",
tpmThreshold = 50, qLow = 0.1, qUp = 0.9)
```

---

plotReverseCumulatives

Plotting reverse cumulative number of CAGE tags per CTSS

Description

Creates PDF file with plots of reverse cumulative number of CAGE tags per CTSS for all CAGE
datasets present in the CAGEset object. The plots should be used as help in choosing the parameters
for power-law normalization: the range of values to fit the power-law and the slope of the referent
power-law distribution (Balwierz *et al*., Genome Biology 2009).

Usage

```
plotReverseCumulatives(object, values = "raw",
fitInRange = c(10,1000), onePlot = FALSE)
```

Arguments

object	A CAGEset object
values	Specifies which values should be plotted. Can be either "raw" to plot reverse cumulatives of raw CAGE tag counts or "normalized" to plot normalized tag count values.
fitInRange	An integer vector with two values specifying a range of tag count values to be used for fitting a power-law distribution to reverse cumulatives. Used only when values = "raw", otherwise ignored. See Details.
onePlot	Logical, should all CAGE datasets be plotted in the same plot (TRUE) or in separate plots (FALSE) within the same PDF file.

Details

Number of CAGE tags (X-axis) is plotted against the number of TSSs that are supported by >= of that number of tags (Y-axis) on a log-log scale for each sample. In addition, a power-law distribution is fitted to each reverse cumulative using the values in the range specified by `fitInRange` parameter. The fitted distribution is defined by `y = -1 * alpha * x + beta` on the log-log scale, and the value of `alpha` for each sample is shown on the plot. In addition, a suggested referent power-law distribution to which all samples should be normalized is drawn on the plot and corresponding parameters (slope alpha and total number of tags T) are denoted on the plot. Referent distribution is chosen so that its slope (alpha) is the median of slopes fitted to individual samples and its total number of tags (T) is the power of 10 nearest to the median number of tags of individual samples. Resulting plots are helpful in deciding whether power-law normalization is appropriate for given samples and reported `alpha` values aid in choosing optimal `alpha` value for referent power-law distribution to which all samples will be normalized. For details about normalization see [normalizeTagCount](#) function.

Value

Creates PDF file named "CTSS_reverse_cumulatives_*_all_samples.pdf" in the working directory, where * denotes either "raw" or "normalized" depending on specified `values` parameter. The file contains plots of reverse cumulative number of CAGE tags per CTSS for each CAGE dataset within CAGEset object. Alpha values of fitted power-laws and suggested referent power-law distribution are reported on the plot in case `values = "raw"`.

Author(s)

Vanja Haberle

References

Balwierz *et al*. (2009) Methods for analyzing deep sequencing expression data: constructing the human and mouse promoterome with deepCAGE data, *Genome Biology* **10**(7):R79.

See Also

[normalizeTagCount](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

plotReverseCumulatives(exampleCAGEset, values = "raw",
fitInRange = c(10,500), onePlot = TRUE)
```

---

quantilePositions   *Determining positions of CAGE signal quantiles within genomic re-*
                           gion

Description

Calculates positions of quantiles of CAGE signal along tag clusters or consensus clusters in each
CAGE dataset within CAGEset object. The function calculates positions of both "lower" and "up-
per" quantiles as described in Details.

Usage

```
quantilePositions(object, clusters, qLow = 0.1, qUp = 0.9,
                  useMulticore = FALSE, nrCores = NULL)
```

Arguments

object	A [CAGEset](#) object
clusters	Which clusters should be used. Can be either `"tagClusters"` to calculate posi-tions of quantiles in tag clusters (different set of genomic coordinates for every CAGE experiment) or `"consensusClusters"` to calculate positions of quan-tiles in consensus clusters (same set of genomic coordinates for every CAGE experiment).
qLow	Which "lower" quantiles should be calculated. It has to be a numeric vector of values in range [0,1]. See Details.
qUp	Which "upper" quantiles should be calculated. It has to be a numeric vector of values in range [0,1]. See Details.
useMulticore	Logical, should multicore be used. `useMulticore = TRUE` is supported only on Unix-like platforms.
nrCores	Number of cores to use when `useMulticore = TRUE`. Default value `NULL` uses all detected cores.

Details

Position of the "lower" quantile qLow is defined as a point that divides the genomic region into two
parts, so that the 5' part contains < qLow * 100% of the CAGE signal of that region. Accordingly,
position of the "upper" quantile qUp is defined as a point that divides the genomic region into two
parts so that the 5' part contains >= qUp * 100% of the CAGE signal of that region. Positions of
one "lower" and one "upper" quantile (when qLow <= qUp) define a central part of the genomic
region that contains >= (qUp - qLow) * 100% of the CAGE signal of that region. Width of that
central part is refered to as "interquantile width", which is a more robust measure of the promoter
width than the total span of the region.

Value

When clusters = "tagClusters", the slots tagClustersQuantileLow and tagClustersQuantileUp of the provided [CAGEset](#) object will be occupied with the positions of specified quantiles in all tag clusters for all CAGE datasets. When clusters = "consensusClusters" the slots consensusClustersQuantileLow and consensusClustersQuantileUp will be occupied by the corresponding information for consensus clusters.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

quantilePositions(object = exampleCAGEset, clusters = "tagClusters",
qLow = c(0.1,0.2), qUp = c(0.8,0.9))
```

---

sampleLabels   *Extracting CAGE datasets labels from CAGEset object*

Description

Extracts the labels of CAGE datasets (samples, experiments) from a CAGEset object.

Usage

```
sampleLabels(object)
```

Arguments

object   A [CAGEset](#) object

Value

Returns a character vector of labels of all CAGE datasets present in the CAGEset object.

Author(s)

Vanja Haberle

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

sampleLabels(exampleCAGEset)
```

---

scoreShift   *Calculating promoter shifting score*

Description

Calculates shifting score for all consensus clusters (promoters) between two specified (groups of) CAGE datasets. Shifting score is a measure of differential usage of TSSs within consensus cluster between two samples, which indicates the degree of physical separation of TSSs used in these samples within given consensus cluster. In addition to shifting score, a statistical significance (P-value and FDR) of differential TSS usage is calculated for each consensus cluster using Kolmogorov-Smirnov test.

Usage

```
scoreShift(object, groupX, groupY, testKS = TRUE, useTpmKS = TRUE, useMulticore = F, nrCores = NULL)
```

Arguments

object	A [CAGEset](#) object
groupX, groupY	Character vector of the one or more CAGE dataset labels in the first (groupX) and in the second group (groupY). Shifting score for each consensus cluster will be calculated by comparing CAGE signal in the samples from groupX against the signal in the samples from groupY. If there is more than one CAGE dataset in the group, the datasets within that group will be merged together before comparison with the other group. See Details.
testKS	Logical, should Kolomogorov-Smirnov test for statistical significance of differential TSS usage be performed, and P-values and FDR returned. See Details.
useTpmKS	Logical, should normalized (tpm) values (TRUE) or raw tag counts (FALSE) be used to derive sample sizes for Kolomogorov-Smirnov test. Used only when testKS = TRUE, otherwise ignored. See Details.
useMulticore	Logical, should multicore be used. useMulticore = TRUE is supported only on Unix-like platforms.
nrCores	Number of cores to use when useMulticore = TRUE. Default value NULL uses all detected cores.

Details

TSSs within one consensus cluster (promoter) can be used differently in different samples (cell types, tissues, developmental stages), with respect to their position and frequency of usage detected by CAGE. This function calculates shifting scores of all consensus clusters between two specified (groups of) CAGE samples to detect promoters that are used differently in these two samples. Shifting score is a measure of differential TSS usage defined as:

```
score = max(F1 - F2) / max(F1)
```

where F1 is a cumulative sum of CAGE signal along consensus cluster in the group of samples

with lower total signal in that consensus cluster, and F2 in the opposite group. Since cumulative sum can be calculated in both forward (5' -> 3') and reverse (3' -> 5') direction, shifting score is calculated for both cases and the bigger value is selected as final shifting score. Value of the shifting score is in the range [-Inf, 1], where value of 1 means complete physical separation of TSSs used in the two samples for given consensus cluster. In general, any non-negative value of the shifting score can be interpreted as the proportion of transcription initiation in the sample with lower expression that is happening "outside" (either upstream or downstream) of the region used for transcription initiation in the other sample. Negative values indicate no physical separation, *i.e.* the region used for transcription initiation in the sample with lower expression is completely contained within the region used for transcription initiation in the other sample.

In addition to shifting score which indicates only physical separation (upstream or downstream shift of TSSs), a more general assessment of differential TSS usage can be obtained by performing a two-sample Kolmogorov-Smirnov test on cumulative sums of CAGE signal along the consensus cluster. In that case, cumulative sums in both samples are scaled to range [0,1] and are considered to be empirical cumulative distribution functions (ECDF) reflecting sampling of TSS positions during transcription initiation. Kolmogorov-Smirnov test is performed to assess whether the two underlying probability distributions differ. To obtain P-value (*i.e.* the level at which the null-hypothesis can be rejected), sample sizes that generated the ECDFs are required, in addition to actual K-S statistics calculated from ECDFs. These are derived either from raw tag counts, *i.e.* exact number of times each TSS in the cluster was sampled during sequencing (when useTpmKS = FALSE), or from normalized tpm values (when useTpmKS = TRUE). P-values obtained from K-S tests are further adjusted for multiple testing using Benjamini & Hochberg (BH) method and for each P-value a corresponding false-discovery rate (FDR) is also reported.

Since calculation of shifting scores and Kolmogorov-Smirnov test require cumulative sums along consensus clusters, they have to be calculated beforehand by calling cumulativeCTSSdistribution function.

Value

The slots shiftingGroupX, shiftingGroupY and consensusClustersShiftingScores of the provided CAGEset object will be occupied by the information on the groups of CAGE datasets that have been compared and shifting scores of all consensus clusters. Consensus clusters (promoters) with shifting score and/or FDR above specified threshold can be extracted by calling getShiftingPromoters function.

Author(s)

Vanja Haberle

See Also

cumulativeCTSSdistribution
getShiftingPromoters

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

scoreShift(object = exampleCAGEset, groupX = c("sample1", "sample2"),
groupY = "sample3", testKS = TRUE, useTpmKS = FALSE)
```

---

setColors

Setting colors for samples

Description

Assigns one color to each sample in the CAGEset object. These colors are used in various plots and exported tracks to consistently represent corresponding samples.

Usage

```
setColors(object, colors = NULL)
```

Arguments

object   A [CAGEset](#) object

colors   A character vector of valid color names. For a complete list of valid color names see colors(). Alternatively, it can be a character vector of colors specified in hexadecimal format (*e.g.* "#FF0000" for red). Number of provided colors must match the number of samples in the CAGEset object. Provided colors are assigned to samples according to their ordering in the CAGEset object, *i.e* in the order they are returned by [sampleLabels](#) function.

Value

Assigns one color to each sample in the CAGEset object by setting them as a name attribute of the sampleLabels slot.

Author(s)

Vanja Haberle

See Also

[sampleLabels](#)

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

setColors(exampleCAGEset, colors = c("darkred", "navy", "forestgreen"))
sampleLabels(exampleCAGEset)
```

---

show-methods *Methods for function* show

Description

Methods for function show.

Methods

signature(object = "CAGEset") Displays a CAGEset object in a user-friendly way, giving an overview of its content.

---

tagClusters *Extracting tag clusters (TCs) for individual CAGE experiment from CAGEset object*

Description

Extracts tag clusters (TCs) produced by [clusterCTSS](#) function for a specified CAGE experiment from a CAGEset object.

Usage

```
tagClusters(object, sample, returnInterquantileWidth = FALSE,
qLow = NULL, qUp = NULL)
```

Arguments

object   A [CAGEset](#) object

sample   Label of the CAGE dataset (experiment, sample) for which to extract tag clusters.

returnInterquantileWidth
                Should the interquantile width for each tag cluster be returned.

qLow   Position of which quantile should be used as a left (lower) boundary when calculating interquantile width. Default value NULL results in using the start coordinate of the cluster. Used only when returnInterquantileWidth = TRUE, otherwise ignored.

qUp   Position of which quantile should be used as a right (upper) boundary when calculating interquantile width. Default value NULL results in using the end coordinate of the cluster. Used only when returnInterquantileWidth = TRUE, otherwise ignored.

Value

Returns a data.frame with genomic coordinates, position of dominant TSS, total CAGE signal and
additional information for all TCs from specified CAGE dataset (sample). If returnInterquantileWidth = TRUE,
interquantile width for each TC is also calculated using specified quantile positions and returned in
the data frame.

Author(s)

Vanja Haberle

See Also

clusterCTSS

Examples

```
load(system.file("data", "exampleCAGEset.RData", package="CAGEr"))

TC <- tagClusters(object = exampleCAGEset, sample = "sample2")
head(TC)
```

Index