Package 'MOCHA'

December 6, 2022

Type Package

Title Modeling for Single-Cell Open Chromatin Analysis

Version 0.1.0

Maintainer Imran McGrath <imran.mcgrath@alleninstitute.org>

Description A statistical framework and analysis tool for open chromatin analysis designed specifically for single cell ATAC-seq (Assay for Transposase-Accessible Chromatin) data, after cell type/cluster identification. These novel modules remove unwanted technical variation, identify open chromatin, robustly models repeated measures in single cell data, implement advanced statistical frameworks to model zero-inflation for differential and co-accessibility analyses, and integrate with existing databases and modules for downstream analyses to reveal biological insights. MOCHA provides a statistical foundation for complex downstream analysis to help advance the potential of single cell ATAC-seq for applied studies. Methods for zero-inflated statistics are as described in: Ghazanfar, S., Lin, Y., Su, X. et al. (2020) <doi:10.1038/s41592-020-0885-x>. Pimentel, Ronald Silva, ``Kendall's Tau and Spearman's Rho for Zero-Inflated Data" (2009) <https://scholarworks.wmich.edu/dissertations/721/>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

Depends R (>= 3.5.0)

Imports data.table, plyranges, dplyr, GenomicRanges, RaggedExperiment, MultiAssayExperiment, SummarizedExperiment, stringr, ggbio, wCorr, magrittr, rlang, AnnotationDbi, BiocGenerics, GenomeInfoDb, GenomicFeatures, IRanges, OrganismDbi, S4Vectors, assertthat, biovizBase, ensembldb, ggplot2, ggrepel, matrixStats, methods, qvalue, scales, tidyr, ggridges

Suggests ArchR, RMariaDB, motifmatchr, BiocManager, TxDb.Hsapiens.UCSC.hg38.refGene, org.Hs.eg.db, BSgenome.Hsapiens.UCSC.hg19, withr, knitr, rmarkdown, testthat (>= 3.0.0)

Additional_repositories https://imran-aifi.github.io/drat

biocViews Data analysis, Chromatin Accessibility, Single cell, scATAC, Software, Visualization

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Samir Rachid Zaim [aut, ctb], Mark-Phillip Pebworth [aut, ctb], Imran McGrath [aut, cre], Lauren Okada [aut, ctb], Xiaojun Li [aut, ctb]

Repository CRAN

Date/Publication 2022-12-06 13:42:38 UTC

R topics documented:

.counts_plot_default_theme . . . 3
.gene_plot_theme . . . 3
addAccessibilityShift . . . 3
addMotifSet . . . 4
annotateTiles . . . 5
callOpenTiles . . . 6
differentialsToGRanges . . . 9
exampleBlackList . . . 10
exampleCellColData . . . 10
exampleFragments . . . 10
extractRegion . . . 11
filterCoAccessibleLinks . . . 12
finalModelObject . . . 13
getCellPopMatrix . . . 13
getCoAccessibleLinks . . . 14
getDifferentialAccessibleTiles . . . 15
getPopFrags . . . 17
getSampleTileMatrix . . . 18
GRangesToString . . . 19
plotConsensus . . . 20
plotRegion . . . 21
StringsToGRanges . . . 24
subsetMOCHAObject . . . 24
youden_threshold . . . 25

Index **26**

.counts_plot_default_theme *Default ggplot theme for counts plot*

Description

Default ggplot theme for counts plot

Usage

```
.counts_plot_default_theme
```

Format

An object of class `list` of length 10.

.gene_plot_theme *Common theme for gene plots*

Description

Common theme for gene plots

Usage

```
.gene_plot_theme
```

Format

An object of class `list` of length 5.

addAccessibilityShift *addAccessibilityShift*

Description

`addAccessibilityShift` will add a new condition to the SummarizedExperiment output of extractRegion, which will contain the difference in accessibility between two conditions

Usage

```
addAccessibilityShift(CountSE, foreground, background, assayName = NULL)
```

Arguments

CountSE	The SummarizedExperiment object output from extractRegion
foreground	Group that will be used as the foreground for the subtraction of accessibility
background	Group that will be used as the background for the subtraction of accessibility
assayName	The name given to the new assay that is difference in accessibility between foreground and background.

Value

countSE a SummarizedExperiment containing coverage for the given input cell populations.

Examples

```
## Not run:
# CountSE is a SummarizedExperiment generated by extractRegion()
countSE <- MOCHA::addAccessibilityShift(
  CountSE = CountSE,
  foreground = "Condition1",
  background = "Condition2",
  assayName = "AccessibilityChanges"
)

## End(Not run)
```

---

addMotifSet addMotifSet

Description

addMotifSetIdentify motifs within peakset

Usage

```
addMotifSet(SE_Object, pwms, w = 7, returnObj = TRUE, motifSetName = "Motifs")
```

Arguments

SE_Object	your MOCHA SummarizedExperiment. Requires Genome AnnotationDbi object within the metadata added by getSampleTileMatrix
pwms	a pwms object for the motif database. Either PFMatrix, PFMatrixList, PWMatrix, or PWMatrixLis'
w	the width for motifmatchr
returnObj	if TRUE, return the modified SE_Object with motif set added to metadata (default). If FALSE, return the motifs from motifmatchr.
motifSetName	name of the motifList in the SE_object's metadata if returnObj=TRUE. Default is 'Motifs'.

Value

the modified SE_Object with motifs added to the metadata

Examples

```
## Not run:
# load a curated motif set from library(chromVARmotifs)
# included with ArchR installation
data(human_pwms_v2)
SE_with_motifs <- addMotifSet(
  SE_Object,
  pwms = human_pwms_v2,
  returnObj = TRUE, motifSetName = "Motifs", w = 7
)

## End(Not run)
```

`annotateTiles` `annotateTiles`

Description

`annotateTiles` annotates a set of sample-tile matrices given with gene annotations. Details on TxDb and Org annotation packages and available annotations can be found at Bioconductor: https://bioconductor.org/package

Usage

```
annotateTiles(Obj, TxDb = NULL, Org = NULL, promoterRegion = c(2000, 100))
```

Arguments

`Obj`	A RangedSummarizedExperment generated from getSampleTileMatrix, containing TxDb and Org in the metadata. This may also be a GRanges object.
`TxDb`	The annotation package for TxDb object for your genome. Optional, only required if Obj is a GRanges.
`Org`	The genome-wide annotation for your organism. Optional, only required if Obj is a GRanges.
`promoterRegion`	Optional list containing the window size in basepairs defining the promoter region. The format is (upstream, downstream). Default is (2000, 100).

Value

Obj, the input data structure with added gene annotations (whether GRanges or SampleTileObj)

Examples

```
## Not run:
library(TxDb.Hsapiens.UCSC.hg38.refGene)
library(org.Hs.eg.db)
SampleTileMatricesAnnotated <- MOCHA::annotateTiles(
  SampleTileMatrices,
  TxDb = TxDb.Hsapiens.UCSC.hg38.refGene,
  Org = org.Hs.eg.db
)

## End(Not run)
```

`callOpenTiles` `callOpenTiles` *Perform peak-calling on a set of fragments or an ArchR Project.*

Description

`callOpenTiles` is the main peak-calling function in MOCHA that serves as a wrapper function to call peaks provided a set of fragment files and an ArchR Project for meta-data purposes

Usage

```
callOpenTiles(
  ATACFragments,
  cellColData,
  blackList,
  genome,
  cellPopLabel,
  cellPopulations = "ALL",
  studySignal = NULL,
  TxDb,
  Org,
  outDir,
  fast = FALSE,
  numCores = 30,
  verbose = FALSE,
  force = FALSE
)

## S4 method for signature 'GRangesList'
callOpenTiles(
  ATACFragments,
  cellColData,
  blackList,
  genome,
```

```
  cellPopLabel,
  cellPopulations = "ALL",
  studySignal = NULL,
  TxDb,
  Org,
  outDir,
  numCores = 30,
  verbose = FALSE,
  force = FALSE
)

## S4 method for signature 'list'
callOpenTiles(
  ATACFragments,
  cellColData,
  blackList,
  genome,
  cellPopLabel,
  cellPopulations = "ALL",
  studySignal = NULL,
  TxDb,
  Org,
  outDir,
  numCores = 30,
  verbose = FALSE,
  force = FALSE
)

.callOpenTiles_ArchR(
  ATACFragments,
  cellPopLabel,
  cellPopulations = "ALL",
  studySignal = NULL,
  TxDb,
  Org,
  outDir = NULL,
  fast = FALSE,
  numCores = 30,
  verbose = FALSE,
  force = FALSE
)
```

Arguments

ATACFragments	an ArchR Project, or a GRangesList of fragments
cellColData	A DataFrame containing cell-level metadata and a 'Sample' column
blackList	A GRanges of blacklisted regions
genome	A valid BSGenome object describing the genome of your organism

cellPopLabel	string indicating which column in the ArchRProject metadata contains the cell population label.
cellPopulations	vector of strings. Cell subsets for which to call peaks. This list of group names must be identical to names that appear in the ArchRProject metadata. Optional, if cellPopulations='ALL', then peak calling is done on all cell populations in the ArchR project metadata. Default is 'ALL'.
studySignal	The median signal (number of fragments) in your study. If not set, this will be calculated using the input ArchR project but relies on the assumption that the ArchR project encompasses your whole study (i.e. is not a subset).
TxDb	is an AnnotationDbi object with transcript info for the organism.
Org	is the genome-wide annotation package for your organism.
outDir	is a string describing the output directory for coverage files and TxDb/Org. Must be a complete directory string. With ArchR input, set outDir to NULL to create a directory within the input ArchR project directory named MOCHA for saving files.
fast	Optional, set to TRUE to use a faster but more memory-intensive
numCores	integer. Number of cores to parallelize peak-calling across multiple cell populations.
verbose	Set TRUE to display additional messages. Default is FALSE.
force	Optional, whether to force creation of coverage files if they already exist. Default is FALSE.

Value

tileResults A MultiAssayExperiment object containing ranged data for each tile

Examples

```
## Not run:
# Starting from an ArchR Project:
library(TxDb.Hsapiens.UCSC.hg38.refGene)
library(org.Hs.eg.db)
tileResults <- MOCHA::callOpenTiles(
  ArchRProj = myArchRProj,
  cellPopLabel = "celltype_labeling",
  cellPopulations = "CD4",
  TxDb = TxDb.Hsapiens.UCSC.hg38.refGene,
  Org = org.Hs.eg.db,
  numCores = 1
)

## End(Not run)

# Starting from GRangesList
if (
  require(BSgenome.Hsapiens.UCSC.hg19) &&
  require(TxDb.Hsapiens.UCSC.hg38.refGene) &&
```

```
  require(org.Hs.eg.db)
) {
tiles <- MOCHA::callOpenTiles(
  ATACFragments = MOCHA::exampleFragments,
  cellColData = MOCHA::exampleCellColData,
  blackList = MOCHA::exampleBlackList,
  genome = BSgenome.Hsapiens.UCSC.hg19,
  TxDb = TxDb.Hsapiens.UCSC.hg38.refGene,
  Org = org.Hs.eg.db,
  outDir = tempdir(),
  cellPopLabel = "Clusters",
  cellPopulations = c("C2", "C5"),
  numCores = 1
)
}
```

`differentialsToGRanges` *differentialsToGRanges Converts a data.frame matrix to a GRanges, preserving additional columns as GRanges metadata*

Description

`differentialsToGRanges` Converts a data.frame matrix to a GRanges, preserving additional columns as GRanges metadata

Usage

```
differentialsToGRanges(differentials, tileColumn = "Tile")
```

Arguments

`differentials`	a matrix/data.frame with a column tileColumn containing region strings in the format "chr:start-end"
`tileColumn`	name of column containing region strings. Default is "Tile".

Value

a GRanges containing all original information

exampleBlackList *exampleBlackList*

Description

Example input of a blackList extracted from the PBMC_Small dataset consisting of 2k cells and spanning chr1 and 2 (~2-300MB). The data is publicly available with the ArchR package at <https://www.archrproject.com/re

Usage

```
exampleBlackList
```

Format

A GRanges object with 210 ranges and 2 metadata columns

exampleCellColData *exampleCellColData*

Description

Example input of cellColData extracted from the PBMC_Small dataset consisting of 2k cells and spanning chr1 and 2 (~2-300MB). The data is publicly available with the ArchR package at <https://www.archrproject.com/re

Usage

```
exampleCellColData
```

Format

A DataFrame with 2217 rows and 3 columns

exampleFragments *exampleFragments*

Description

Example input of ATAC fragments extracted from the PBMC_Small dataset consisting of 2k cells and spanning chr1 and 2 (~2-300MB). This subset consists of two cell populations: Clusters C2 and C5. The data is publicly available with the ArchR package at <https://www.archrproject.com/reference/getTestProject.html>

Usage

```
exampleFragments
```

Format

A list of 2 GRanges objects

`extractRegion` `extractRegion`

Description

`extractRegion` will extract the coverage files created by callOpenTiles and return a specific region's coverage

Usage

```
extractRegion(
  SampleTileObj,
  region,
  cellPopulations = "ALL",
  groupColumn = NULL,
  subGroups = NULL,
  sampleSpecific = FALSE,
  approxLimit = 1e+05,
  binSize = 250,
  numCores = 1,
  verbose = FALSE
)
```

Arguments

`SampleTileObj`	The SummarizedExperiment object output from getSampleTileMatrix
`region`	a GRanges object or vector or strings containing the regions on which to compute co-accessible links. Strings must be in the format "chr:start-end", e.g. "chr4:1300-2222".
`cellPopulations`	vector of strings. Cell subsets for which to call peaks. This list of group names must be identical to names that appear in the SampleTileObj. Optional, if cellPopulations='ALL', then peak calling is done on all cell populations. Default is 'ALL'.
`groupColumn`	Optional, the column containing sample group labels for returning coverage within sample groups. Default is NULL, all samples will be used.
`subGroups`	a list of subgroup(s) within the groupColumn from the metadata. Optional, default is NULL, all labels within groupColumn will be used.
`sampleSpecific`	If TRUE, get a sample-specific count dataframe out. Default is FALSE, average across samples and get a dataframe out.
`approxLimit`	Optional limit to region size, where if region is larger than approxLimit basepairs, binning will be used. Default is 100000.

binSize	Optional, size of bins in basepairs when binning is used. Default is 250.
numCores	integer. Number of cores to parallelize peak-calling across multiple cell populations
verbose	Set TRUE to display additional messages. Default is FALSE.

Value

countSE a SummarizedExperiment containing coverage for the given input cell populations.

Examples

```
## Not run:
countSE <- MOCHA::extractRegion(
  SampleTileObj = SampleTileMatrices,
  cellPopulations = "ALL",
  region = "chr1:18137866-38139912",
  numCores = 30,
  sampleSpecific = FALSE
)

## End(Not run)
```

---

filterCoAccessibleLinks

filterCoAccessibleLinks

Description

filterCoAccessibleLinks will filter the output from getCoAccessibleLinks by a threshold, retaining links with a absolute correlation greater than the threshold. This function also adds the chr, start, and end site of each link to the output table.

Usage

```
filterCoAccessibleLinks(TileCorr, threshold = 0.5)
```

Arguments

TileCorr	The correlation table output from getCoAccessibleLinks
threshold	Keep

Value

FilteredTileCorr The filtered correlation table with chr, start, and end site of each link

Examples

```
## Not run:
# links is the output of MOCHA::getCoAccessibleLinks
MOCHA::filterCoAccessibleLinks(links, threshold = 0.5)

## End(Not run)
```

`finalModelObject` *finalModelObject*

Description

Trained MOCHA models - LOESS and linear regression

Usage

```
finalModelObject
```

Format

A list of lists containing 2 items: "Loess" and "Linear" each with "Total" "Max" and "Intercept"

Loess LOESS model

Linear Linear model

`getCellPopMatrix` `getCellPopMatrix`

Description

`getCellPopMatrix` pulls out the SampleTileMatrix of tiles called in one given cell population.

Usage

```
getCellPopMatrix(
  SampleTileObj,
  cellPopulation,
  dropSamples = TRUE,
  NAtoZero = TRUE
)
```

Arguments

SampleTileObj	The output from getSampleTileMatrix, a SummarizedExperiment of pseudobulk intensities across all tiles & cell types.
cellPopulation	The cell population you want to pull out.
dropSamples	Boolean flag to determine whether to drop samples that were too small for peak calling.
NAtoZero	Boolean flag to determine whether to replace NAs with zero

Value

sampleTileMatrix a matrix of samples by called tiles for a given cell population.

---

getCoAccessibleLinks getCoAccessibleLinks

Description

getCoAccessibleLinks takes an input set of regions (tiles) and finds co-accessible neighboring regions within a window. Co-accessibility is defined as the correlation between two region intensity (openness) across samples.

Usage

```
getCoAccessibleLinks(
  SampleTileObj,
  cellPopulation = "All",
  regions,
  windowSize = 1 * 10^6,
  numCores = 1,
  ZI = TRUE,
  verbose = FALSE
)
```

Arguments

SampleTileObj	The SummarizedExperiment object output from getSampleTileMatrix containing your sample-tile matrices
cellPopulation	A string denoting the cell population of interest, which must be present in SampleTileObj
regions	a GRanges object or vector or strings containing the regions on which to compute co-accessible links. Strings must be in the format "chr:start-end", e.g. "chr4:1300-2222". Can be the output from getDifferentialAccessibleTiles.
windowSize	the size of the window, in basepairs, around each input region to search for co-accessible links
numCores	Optional, the number of cores to use with multiprocessing. Default is 1.

ZI	boolean flag that enables zero-inflated (ZI) Spearman correlations to be used. Default is TRUE. If FALSE, skip zero-inflation and calculate the normal Spearman.
verbose	Set TRUE to display additional messages. Default is FALSE.

Details

The technical details of the zero-inflated correlation can be found here:

Pimentel, Ronald Silva, "Kendall's Tau and Spearman's Rho for Zero-Inflated Data" (2009). Dissertations.

while the implementation (scHOT R package), can be found here: http://www.bioconductor.org/packages/release/bioc/html/sc

Value

TileCorr A data.table correlation matrix

`getDifferentialAccessibleTiles`
`getDifferentialAccessibleTiles`

Description

`getDifferentialAccessibleTiles` allows you to determine whether regions of chromatin are differentially accessible between groups by conducting a test

Usage

```
getDifferentialAccessibleTiles(
  SampleTileObj,
  cellPopulation,
  groupColumn,
  foreground,
  background,
  signalThreshold = 12,
  minZeroDiff = 0.5,
  fdrToDisplay = 0.2,
  outputGRanges = TRUE,
  numCores = 2,
  verbose = FALSE
)
```

Arguments

SampleTileObj	The SummarizedExperiment object output from getSampleTileMatrix
cellPopulation	A string denoting the cell population of interest
groupColumn	The column containing sample group labels

foreground	The foreground group of samples for differential comparison
background	The background group of samples for differential comparison
signalThreshold	Minimum median intensity required to keep tiles for differential testing to increase statistical power in small sample cohorts. Default is 12.
minZeroDiff	Minimum difference in average dropout rates across groups require to keep tiles for differential testing. Default is 0.5 (50%).
fdrToDisplay	False-discovery rate used only for standard output messaging. Default is 0.2.
outputGRanges	Outputs a GRanges if TRUE and a data.frame if FALSE. Default is TRUE.
numCores	The number of cores to use with multiprocessing. Default is 1.
verbose	Set TRUE to display additional messages. Default is FALSE.

Value

full_results The differential accessibility results as a GRanges or matrix data.frame depending on the flag 'outputGRanges'.

Examples

```
## Not run:
cellPopulation <- "MAIT"
foreground <- "Positive"
background <- "Negative"
# Standard output will display the number of tiles found below a false-discovery rate threshold.
# This parameter does not filter results and only affects the aforementioned message.
fdrToDisplay <- 0.2
# Choose to output a GRanges or data.frame.
# Default is TRUE
outputGRanges <- TRUE
# SampleTileMatrices is the output of MOCHA::getSampleTileMatrix
differentials <- MOCHA::getDifferentialAccessibleTiles(
  SampleTileObj = SampleTileMatrices,
  cellPopulation = cellPopulation,
  groupColumn = groupColumn,
  foreground = foreground,
  background = background,
  fdrToDisplay = fdrToDisplay,
  outputGRanges = outputGRanges,
  numCores = numCores
)

## End(Not run)
```

getPopFrags *Extract fragments by populations from an ArchR Project*

Description

getPopFrags returns a list of fragments per cell subset as a GRanges.

Usage

```
getPopFrags(
  ArchRProj,
  metaColumn,
  cellSubsets = "ALL",
  region = NULL,
  numCores = 1,
  sampleSpecific = TRUE,
  NormMethod = "nfrags",
  blackList = NULL,
  verbose = FALSE,
  overlapList = 50
)
```

Arguments

ArchRProj	The ArchR Project.
metaColumn	The name of metadata column that contains the populations of cells you want to merge and export.
cellSubsets	Default is 'ALL'. If you want to export only some groups, then give it a list of group names. This needs to be unique - no duplicated names. This list of group names must be identical to names that appear in the metadata column of the ArchR Project (e.g. metaColumn).
region	Optional parameter. Set this if you only want to extract fragments from particular regions of the genome. Format should be as a string (e.g. 'chr1:1000-2000'), or a GRanges object.
numCores	Number of cores to use.
sampleSpecific	Set to TRUE to further subset cells by sample
NormMethod	Normalization method. Can be either "nFrags","nCells", or "Median".
blackList	Blacklisted region to filter out. Default is to not filter out anything (i.e. NULL). Input should be provided as a GRanges object. Any fragments with more than a certain overlap will be thrown out.
verbose	Set TRUE to display additional messages. Default is FALSE.
overlapList	The minimum overlap necessary for a fragment marked as overlapping with the blacklist region and thus thrown out.

Value

A list of GRanges containing fragments. Each GRanges corresponds to a population defined by cellSubsets (and sample, if `sampleSpecific=TRUE`)

`getSampleTileMatrix` `getSampleTileMatrix`

Description

`getSampleTileMatrix` takes the output of peak calling with callOpenTiles and creates sample-tile matrices containing the signal intensity at each tile.

Usage

```
getSampleTileMatrix(
  tileResults,
  cellPopulations = "ALL",
  groupColumn = NULL,
  threshold = 0.2,
  log2Intensity = TRUE,
  numCores = 1,
  verbose = FALSE
)
```

Arguments

`tileResults`	a MultiAssayExperiment returned by callOpenTiles containing containing peak calling results.
`cellPopulations`	vector of strings. Cell subsets in TileResults for which to generate sample-tile matrices. This list of group names must be identical to names that appear in the ArchRProject metadata. If cellPopulations='ALL', then peak calling is done on all cell populations in the ArchR project metadata. Default is 'ALL'.
`groupColumn`	Optional, the column containing sample group labels for determining consensus tiles within sample groups. Default is NULL, all samples will be used for determining consensus tiles.
`threshold`	Threshold for consensus tiles, the minimum % of samples (within a sample group, if groupColumn is set) that a peak must be called in to be retained. If set to 0, retain the union of all samples' peaks (this is equivalent to a threshold of 1/numSamples). It is recommended to tune this parameter to omit potentially spurious peaks.
`log2Intensity`	Boolean, set to TRUE to return the log2 of the sample-tile intensity matrix. Optional, default is FALSE.
`numCores`	Optional, the number of cores to use with multiprocessing. Default is 1.
`verbose`	Set TRUE to display additional messages. Default is FALSE.

Value

SampleTileMatrices a MultiAssayExperiment containing a sample-tile intensity matrix for each cell population

Examples

```
# Starting from GRangesList
if (
  require(BSgenome.Hsapiens.UCSC.hg19) &&
  require(TxDb.Hsapiens.UCSC.hg38.refGene) &&
  require(org.Hs.eg.db)
) {
tiles <- MOCHA::callOpenTiles(
  ATACFragments = MOCHA::exampleFragments,
  cellColData = MOCHA::exampleCellColData,
  blackList = MOCHA::exampleBlackList,
  genome = BSgenome.Hsapiens.UCSC.hg19,
  TxDb = TxDb.Hsapiens.UCSC.hg38.refGene,
  Org = org.Hs.eg.db,
  outDir = tempdir(),
  cellPopLabel = "Clusters",
  cellPopulations = c("C2", "C5"),
  numCores = 1
)

SampleTileMatrices <- MOCHA::getSampleTileMatrix(
  tiles,
  cellPopulations = c('C2', 'C5'),
  threshold = 0 # Take union of all samples' open tiles
)
}
```

`GRangesToString` `GRangesToString` *Converts a GRanges object to a string in the format 'chr1:100-200'*

Description

`GRangesToString` Turns a GRanges Object into a list of strings in the format chr1:100-200

Usage

```
GRangesToString(GR_obj)
```

Arguments

`GR_obj`	the GRanges object to convert to a string

Value

A string or list of strings in the format 'chr1:100-200' representing ranges in the input GRanges

---

`plotConsensus` `plotConsensus`

Description

`plotConsensus` Extracts the peak reproducibility and generates a heuristic plots that can be used to determine the reproducibility threshold used within getSampleTileMatrix.

Usage

```
plotConsensus(
  tileObject,
  cellPopulations = "All",
  groupColumn = NULL,
  returnPlotList = FALSE,
  returnDFs = FALSE,
  numCores = 1
)
```

Arguments

`tileObject`	A MultiAssayExperiment object from callOpenTiles,
`cellPopulations`	the cell populations you want to visualize.
`groupColumn`	Optional parameter, same as in getSampleTileMatrix, which defines whether you want to plot reproducibility within each
`returnPlotList`	Instead of one plot with all celltypes/conditions, it returns a list of plots for each cell types
`returnDFs`	Instead of a plot, returns a data.frame of the reproducibility across samples. If set to false, then it plots the data.frame instead of returning it.
`numCores`	Number of cores to multithread over.

Value

SampleTileObj the input data structure with added gene annotations.

plotRegion plotRegion

Description

plotRegion Plots the region that you've summarized across all cell groupings (groups=initial getPopFrags() split) with optional motif overlay, chromosome position ideogram, and additional GRanges tracks. If plotting motif overlay, ensure that motif annotations have been added to your counts SummarizedExperiment. A basic plot can be rendered with just a counts SummarizedExperiment, but additional formatting arguments allow for further customization. Note that to show specific genes with the option 'whichGene' the **RMariaDB** package must be installed.

Usage

```
plotRegion(
  countSE,
  plotType = "area",
  base_size = 12,
  counts_color = NULL,
  range_label_size = 2,
  legend.position = NULL,
  facet_label_side = "top",
  counts_color_var = "Groups",
  counts_group_colors = NULL,
  counts_theme_ls = NULL,
  motifSetName = NULL,
  motif_y_space_factor = 4,
  motif_stagger_labels_y = FALSE,
  motif_weights = NULL,
  motif_weight_name = "Motif Weight",
  motif_weight_colors = c(darkblue = -10, gray = 0, darkred = 10),
  motif_lab_size = 1,
  motif_lab_alpha = 0.25,
  motif_line_alpha = 0.25,
  motif_line_size = 0.75,
  showGene = TRUE,
  whichGene = NULL,
  db_id_col = "REFSEQ",
  collapseGenes = "None",
  gene_theme_ls = NULL,
  additionalGRangesTrack = NULL,
  linkdf = NULL,
  showIdeogram = TRUE,
  ideogram_genome = "hg19",
  relativeHeights = c(Chr = 0.9, `Normalized Counts` = 7, Links = 1.5, Genes = 2,
    AdditionalGRanges = 4.5),
  verbose = FALSE
```

)

Arguments

countSE	A SummarizedExperiment from MOCHA::getCoverage
plotType	Options include 'overlaid','area', or 'RidgePlot'. default is 'area', which will plot a separate track for each group with the area filled in under the curve. Setting plotType to 'overlaid' will overlay count plot histograms across samples, instead of faceting out separately. Setting plotType to 'RidgePlot' will generate a ridgeplot across all groups.
base_size	Numeric, default 12. Global plot base text size parameter
counts_color	Optional color palette. A named vector of color values where names are unique values in the 'color_var' column
range_label_size	Numeric value, default 4. Text size for the y-axis range label
legend.position	Any acceptable 'legend.position' argument to theme(). Default NULL will place legend for overlaid plots at (0.8,0.8), or to the "right" for faceted plots.
facet_label_side	Direction character value, default "top". Can also be "right", "left", or "bottom". Position of facet label.
counts_color_var	Character value, default "Groups". Column name from countdf to use to color counts plots. Only used if counts_group_colors provided
counts_group_colors	Optional named color vector. Values as colors, names are levels of 'counts_color_var'. If provided, will color the plots specifically using 'scale_color_manual()'
counts_theme_ls	A list of named theme arguments passed to theme(). For example, 'list(axis.ticks = element_blank())'. Default NULL will use '.counts_plot_default_theme'.
motifSetName	The name of the motif set in ArchRProj to use for annotation. Example: 'JasparMotifs'
motif_y_space_factor	A factor for vertical spacing between motif labels. Default 4. Increase to make labels farther apart, decrease to make labels closer.
motif_stagger_labels_y	= FALSE Logical value, default FALSE. If TRUE, will stagger motif labels in adjacent columns in the vertical direction
motif_weights	Optional numeric vector, default NULL. If provided will be used to color motif labels by the weighted values
motif_weight_name	Character value, default "Motif Weight". Used to label the legend for motif colors
motif_weight_colors	Named numeric vector. Names should be color values and breaks should be the corresponding values of motif_weights. Values outside the highest and lowest value will appear as max or min defined color value.

motif_lab_size Numeric value, default 1. Size of motif labels.

motif_lab_alpha
: Numeric value, default 0.25. Alpha for motif labels.

motif_line_alpha
: Numeric value, default 0.25. Alpha for motif lines.

motif_line_size
: Numeric value, default 1. Size of motif lines.

showGene Logical value, default TRUE. Whether or not the gene track should be plotted.

whichGene Name of gene for plotting this specific gene in region.

db_id_col Character value. Column in ‘orgdb‘ containing the output id for ‘whichGene‘ plotting. Default "REFSEQ".

collapseGenes Options include ’collapseAll’, ’longestTx’, or ’None’ Default ’None’ will plot the expanded view of the reference genes, ’collapseAll’ if you want collapse the gene tracks into one, and ’longestTx’ will only plot the longest transcript of each gene.

gene_theme_ls Named list of parameters passed to ‘theme()‘ for the gene plot. Default NULL will use ‘.gene_plot_theme‘

additionalGRangesTrack
: A GRanges object containing additional track plot data

linkdf A dataframe with co-accessible links to display as an additional track

showIdeogram Logical value, default TRUE. If TRUE plots the chromosome ideogram at the top of the multi-track plot

ideogram_genome
: Character value, a genome name for the ideogram plot. Default ’hg19’.

relativeHeights
: Named numeric vector of relative heights for each of the 4 track plots to enable clean visualization when there are many tracks. Unused tracks will be ignored. Default value = c(‘Chr‘ = 0.9, ‘Normalized Counts‘ = 7, ‘Genes‘= 2, ‘AdditionalGRanges‘ = 4.5)

verbose Set TRUE to display additional messages. Default is FALSE.

Value

The input ggplot object with motif labels overlaid

Examples

```
## Not run:
# my_count_SE is a counts data frame generated by extractRegion()

# Simple counts + ideogram + all genes:
plotRegion(countSE = my_count_SE)

# Motif overlay for a project my_proj containing "JasparMotifs" annotations:
plotRegion(
  countSE = my_count_SE, motifSetName = "JasparMotifs",
```

```
  motif_lab_alpha = 1, motif_line_alpha = 1
)

# Motif overlay w/ weights:
plotRegion(
  countSE = my_count_SE, motifSetName = "JasparMotifs", motif_lab_alpha = 1,
  motif_line_alpha = 1, motif_weights = my_enrichment_weights
)

## End(Not run)
```

StringsToGRanges StringsToGRanges

Description

StringsToGRanges Turns a list of strings in the format chr1:100-200 into a GRanges object

Usage

```
StringsToGRanges(regionString)
```

Arguments

regionString	A string or list of strings each in the format chr1:100-200

Value

a GRanges object with ranges representing the input string(s)

---

subsetMOCHAObject subsetObject

Description

subsetObject subsets a tileResults-type object (from callOpenTiles), or a SummarizedExperiment-type object (from getSampleTileMatrix), either by cell type or sample metadata.

Usage

```
subsetMOCHAObject(Object, subsetBy, groupList, na.rm = TRUE, verbose = FALSE)
```

Arguments

Object	A MultiAssayExperiment or RangedSummarizedExperiment,
subsetBy	the variable to subset by. Can either be 'celltype', or a column from the sample metadata (see colData(Object))
groupList	the list of cell type names or sample-associated data that should be used to subset the Object
na.rm	removes groups that are NA if set to true. If set to false, then you filter for everything in the groupList and also NA values.
verbose	Set TRUE to display additional messages. Default is FALSE.

Value

Object the input Object, filtered down to either the cell type or samples desired.

---

youden_threshold *youden_threshold*

Description

Trained regression model for predicting a cutoff threshold for peak calling. Call: loess(formula = OptimalCutpoint ~ Ncells, data = thresh_df)

Usage

```
youden_threshold
```

Format

A list of 18 regression variables

Details

Number of Observations: 27 Equivalent Number of Parameters: 5.98 Residual Standard Error: 0.02121

Index

∗ **datasets**
    .counts_plot_default_theme, 3
    .gene_plot_theme, 3
    exampleBlackList, 10
    exampleCellColData, 10
    exampleFragments, 10
    finalModelObject, 13
    youden_threshold, 25
.callOpenTiles_ArchR *(*callOpenTiles*)*, 6
.counts_plot_default_theme, 3
.gene_plot_theme, 3

addAccessibilityShift, 3
addMotifSet, 4
annotateTiles, 5

callOpenTiles, 6
callOpenTiles,ArchRProject-method
        *(*callOpenTiles*)*, 6
callOpenTiles,GRangesList-method
        *(*callOpenTiles*)*, 6
callOpenTiles,list-method
        *(*callOpenTiles*)*, 6

differentialsToGRanges, 9

exampleBlackList, 10
exampleCellColData, 10
exampleFragments, 10
extractRegion, 11

filterCoAccessibleLinks, 12
finalModelObject, 13

getCellPopMatrix, 13
getCoAccessibleLinks, 14
getDifferentialAccessibleTiles, 15
getPopFrags, 17
getSampleTileMatrix, 18
GRangesToString, 19

plotConsensus, 20
plotRegion, 21

StringsToGRanges, 24
subsetMOCHAObject, 24

youden_threshold, 25